Pruebas de conocimiento cero

Doriana Hernández Fernández
Trabajo de Fin de Grado

Dirigido por
José Ignacio Iglesias Curto
Francisco José Plaza Martín

Grado en Matemáticas
Facultad de Ciencias

Julio 2020

Pruebas de conocimiento cero

Doriana Hernández Fernández
Trabajo de Fin de Grado

Dirigido por
José Ignacio Iglesias Curto
Francisco José Plaza Martín

Grado en Matemáticas
Facultad de Ciencias

Julio 2020

Índice general

Capítulo 1

Introducción

Las pruebas de conocimiento cero son protocolos criptográficos en los que participan dos entidades o participantes: el probador y el verificador. El probador debe demostrar al verificador que conoce una información sin revelarla.

Las áreas en las que va a centrarse el trabajo serán álgebra, en particular, criptografía, y probabilidad.

La aparición de las pruebas de conocimiento cero supone un planteamiento nuevo, puesto que no se transmite la información en sí, sino una garantía de que se conoce esa información. En otros métodos de encriptación se ocultaba la información que se quería mantener en secreto y existía la posibilidad de acceder a la información rompiendo el protocolo.

La primera mención a las pruebas de conocimiento cero la realizan Goldwasser, Micali y Rackoff en 1985 [10], presentan las pruebas de conocimiento cero como una prueba interactiva para demostrar la pertenencia de un elemento a un lenguaje sin demostrar nada más que la pertenencia.

Posteriormente, en 1987, Fiat y Shamir presentan un protocolo de identificación [8] utilizando la idea de prueba de conocimiento cero interactiva de Goldwasser, Micali y Rackoff. Aquí, se pasa de demostrar la pertenencia de un elemento a un lenguaje, a demostrar la identidad de un participante o entidad, basándose en el conocimiento de un secreto que solo el probador conoce. Además de presentar un protocolo de identificación, presentan un protocolo de firma digital.

En 1988, Blum, Feldman y Micali presentan las pruebas de conocimiento cero no interactivas [3] que mejoran la idea de conocimiento cero haciéndola no interactiva, ya que en algunos contextos no era factible. Estas pruebas vuelven al problema de demostrar que un elemento pertenece a un lenguaje. Bellarre y Goldwasser, recogen la idea de prueba no interactiva de conocimiento cero y construyen un protocolo de firma digital en 1990 [2].

Actualmente, las pruebas de conocimiento cero se aplican en *Blockchain* [13] y, en general, en procesos en los que se necesita mantener un nivel de seguridad alto.

La memoria que se presenta contiene una vista general de las pruebas de conocimiento cero en todas sus formas y variantes, así como una evolución del concepto y sus diferentes adaptaciones. Además, contiene protocolos de este tipo de pruebas junto con ejemplos concretos. El trabajo se divide en seis capítulos: al final del capítulo en el que nos encontramos, ofrecemos un relato ejemplificador de lo que es una prueba de conocimiento cero. En el Capítulo 2, presentamos las pruebas de conocimiento cero, para ello aportamos unas

nociones previas para formalizar la definición de estas pruebas. En el Capítulo 3, mostramos tres casos concretos de pruebas de conocimiento cero sobre lenguajes y el desarrollo completo de uno de ellos en base a las definiciones del Capítulo 2. En el Capítulo 4, pasamos a las pruebas de conocimiento cero para la identificación, se abordan tres pruebas de este tipo y se incluyen ejemplos de las mismas. En el Capítulo 5, se exponen las pruebas de conocimiento cero no interactivas, las firmas digitales y varios ejemplos de ellos. En el Capítulo 6, presentamos una aplicación actual que tienen las pruebas de conocimiento cero y cómo se ha adaptado la idea al *blockchain*. Por último, incluimos una conclusión final del desarrollo del trabajo.

1.1. Ejemplo

Explicaremos una prueba de conocimiento cero utilizando un relato para tener una idea general de lo que es una prueba de este tipo.

Como ya sabemos, en las pruebas de conocimiento cero participan dos entidades: el probador y el verificador, donde el probador demuestra al verificador que conoce una información sin revelarla. En este relato nos encontramos a Paquita, que es el probador, a los periodistas, que son el verificador y la palabra secreta, que es la información que posee el probador y que no quiere que nadie conozca.

Supongamos dos pueblos llamados Macondo y Bree situados en una isla, y que además están separados por una montaña inaccesible. Cada pueblo tiene una cueva sin salida en su lado de la montaña. No existe ningún tipo de conexión entre ambos pueblos porque ningún habitante se atreve a echarse a la mar y porque ninguno es capaz de escalar la montaña.

Paquita es una habitante de uno de estos pueblos, no revela de cuál, y quiere demostrar al mundo que conoce una puerta secreta que conecta ambos pueblos. Esta puerta conectaría las cuevas de cada pueblo y solo se abriría utilizando una palabra secreta que solo Paquita conoce. Como quiere que nadie conozca la palabra secreta se pone en contacto con unos periodistas para demostrarles que conoce la palabra mágica sin decírsela.

Un equipo de periodistas foráneos son los que comprobarán que Paquita conoce la palabra secreta. El protocolo que se sigue es el siguiente: el equipo de periodistas se aleja de la isla y pide a Paquita que entre en la cueva de su pueblo. Cuando Paquita se encuentra dentro de la cueva, avisa a los periodistas que lanzan una moneda: si sale cara piden a Paquita que salga por la entrada de Macondo; si sale cruz, por la de Bree. Después de lanzar la moneda, los periodistas van a la entrada de la cueva correspondiente. Paquita sale por la cueva que le piden, los periodistas vuelven a alejarse y Paquita regresa a su pueblo. Visto así, podría pensarse que si Paquita se encuentra en la cueva de Bree y los periodistas le piden que salga por la entrada de Bree, Paquita no tendría por qué saberse la palabra mágica. Por ello, este proceso se repite varias veces para reducir la probabilidad lo máximo posible de que Paquita salga por la misma cueva que entró.

Al finalizar todo el proceso, Paquita ha demostrado al mundo que existe una puerta mágica que une dos pueblos y que solo ella conoce la palabra que la abre.

Si tenemos presente este ejemplo durante las siguientes páginas nos resultará más fácil comprender las nociones que vamos a dar.

Capítulo 2

Pruebas de conocimiento cero

En este capítulo abordaremos las nociones previas necesarias para poder dar una definición formal de lo que es una prueba de conocimiento cero. Estas nociones incluyen las máquinas de Turing, los protocolos interactivos y los sistemas de pruebas interactivas. Además, para poder definir lo que es el conocimiento cero formalmente, presentaremos el concepto de indistinguibilidad.

2.1. Máquinas de Turing

En esta sección presentamos la definición de máquina de Turing y de las variantes para poder definir con rigor las pruebas de conocimiento cero. La bibliografía sobre este tema era bastante escasa y fue necesario hacer una combinación de toda la información encontrada sobre máquinas de Turing pero en particular de [5] y [6] , y del tipo de máquina de Turing que se necesitaba para formalizar la definición de prueba de conocimiento cero [10].

Los dos participantes de una prueba de conocimiento cero pueden pensarse como dos máquinas de Turing que interactúan entre sí. Esto nos será útil para dar la definición formal de una prueba de conocimiento cero.

Una máquina de Turing es un dispositivo teórico que posee una memoria externa infinita denominada cinta, dividida en celdas; que adopta un estado interno en cada momento; y que contiene unas instrucciones.

Cada una de las celdas contiene un símbolo de un alfabeto o un espacio en blanco (#) y hay un selector que determina qué celda está leyendo la máquina en ese momento. En cuanto a los estados internos, hay dos que son los más especiales: el estado inicial, es decir, el estado interno de la máquina cuando se activa; y el estado final, que es el estado interno al que pasa la máquina y después para. Las instrucciones de la máquina especifican que dados un estado interno y un símbolo leído, entonces se pasa a otro estado interno, otro símbolo y hacia dónde debe moverse el selector de celdas. Podría darse el caso de que el estado interno o el símbolo no cambiaran, así como que el selector se mantuviera en el mismo sitio.

Definición 2.1. Una máquina de Turing es una 6-úpla $(Q, q_0, q_f, S, \#, \delta)$ donde:

- Q es un conjunto finito. Este conjunto contiene todos los estados internos de la máquina.

- q_0 es el estado inicial de la máquina.
- q_f es el estado final de la máquina.
- S es un alfabeto finito. Está formado por los símobolos que ocupan las celdas.
- $\#$ es el espacio en blanco.
- $\delta : Q \setminus \{q_f\} \times S \sqcup \# \to Q \times S \sqcup \# \times \{I, D, N\}$ es una función a la que llamamos función de transición donde $\{I, D, N\}$ hace referencia al movimiento que debe hacer el selector: a la izquierda (I), a la derecha (D) o no moverse (N). Esta función nos determina las instrucciones que sigue la máquina de Turing.

El funcionamiento es el siguiente: la máquina se activa, se encuentra en su estado interno inicial y el selector está en una celda. Con las instrucciones, se pasa a otro estado, se escribe en la celda otro símbolo y se mueve, o no, el selector de celdas. Este proceso se repetiría hasta que llegara al estado final y parase. Podría ocurrir que la máquina entrase en un bucle infinito y no llegara nunca a ese estado final. Las máquinas de Turing que utilizaremos para formalizar la definición de prueba de conocimiento cero no contendrán bucles infinitos.

Definición 2.2. Una máquina de Turing es determinista si para cada par, estado-símbolo, existe una única terna, estado-símbolo-movimiento. En el caso de que uno de esos pares pueda dar más de una terna, entonces decimos que es una máquina de Turing no determinista.

Definición 2.3. Una máquina de Turing multicinta, es una máquina de Turing con n cintas y con n selectores de celdas. Su función de transición será: $\delta : Q \setminus \{q_f\} \times (S \sqcup \#)^n \to Q \times (S \sqcup \#)^n \times \{I, D, N\}^n$

Definición 2.4. Una máquina de Turing probabilística es una máquina de Turing determinista, con una cinta adicional de solo lectura que contiene una cadena binaria aleatoria que solo puede leerse de izquierda a derecha. Aquí, la función de transición δ dependerá también de la lectura de la cinta aleatoria.

Definición 2.5 ([10])**.** Una máquina de Turing interactiva es una máquina de Turing multicinta y probabilística que está formada por seis cintas:

- Una cinta de entrada de solo lectura.
- Una cinta de comunicación de solo lectura.
- Una cinta de comunicación de solo escritura.
- Una cinta aleatoria de solo lectura.
- Una cinta de trabajo de lectura y escritura.
- Una cinta de salida de solo escritura.

Lo más característico de estas máquinas interactivas son las cintas de comunicación, la de escritura contendría los "mensajes"que envía la máquina y la de lectura, los "mensajes"que recibe. Estos mensajes son los que dotan a estas máquinas de Turing de interacción. La cinta aleatoria de solo lectura es la que hace a esta máquina de Turing probabilística. La cinta de entrada contiene el valor de entrada de la máquina con el que luego opera la cinta de trabajo y la cinta de salida contiene el valor de salida. La cinta de trabajo es la cinta que tienen todas las máquinas de Turing por definición.

Definición 2.6 ([10])**.** Un par de máquinas de Turing interactivas son dos máquinas de Turing interactivas A y B que cumplen lo siguiente: comparten la cinta de entrada; la cinta de comunicación de solo lectura de A es la cinta de comunicación de solo escritura de B; y viceversa. Un par de máquinas de Turing interactivas se denota por (A, B).

Antes de pasar a la siguiente sección, vamos a dar unas definiciones basadas en una característica importante de las máquinas de Turing, la complejidad, es decir, el número de pasos que dedica a realizar todas sus tareas hasta que para. Esta complejidad suele medirse en términos de la longitud del valor de entrada de la máquina.

Definición 2.7 ([9])**.** Sea M una máquina de Turing probabilística, denotamos $t_M(x, r)$ al número de pasos que tiene que hacer M con entrada x y con r la secuencia de bits aleatorios hasta parar.

Definición 2.8. $\{0,1\}^*$ es el conjunto de todas las secuencias finitas de 0 y 1. Cuando escribamos $|x|$ estaremos denotando la longitud de x, siendo x una secuencia finita en $\{0,1\}^*$.

Definición 2.9. Un lenguaje es un subconjunto de $\{0,1\}^*$ que se denota por L.

Definición 2.10 ([9])**.** Con las condiciones de la Definición 2.7, la máquina M es de tiempo esperado polinomial si para todo $x \in \{0,1\}^*$ existe un polinomio Q de modo que el número esperado de pasos en función de la secuencia r de $t_M(x, r)$ va a estar acotado por $Q(|x|)$, donde $|x|$ es la longitud de la entrada x. Es decir, el número de pasos que se espera que dé una máquina M está acotado por un polinomio que depende solo de la longitud de su entrada.

2.2. Protocolos y sistemas de pruebas interactivas

Después de definir las máquinas de Turing que vamos a utilizar, pasamos a definir los protocolos interactivos y los sistemas de pruebas interactivas que son la antesala de las pruebas de conocimiento cero.

Definición 2.11. [10] Sea (A, B) un par de máquinas de Turing, (A, B) es un protocolo interactivo si A no tiene límite computacional y B es de tiempo esperado polinomial.

El funcionamiento de un par de máquinas de Turing es el siguiente: el objetivo final es que la máquina B acepte o rechace el valor de entrada del par de máquinas en base al cumplimiento de unas condiciones. Las máquinas están activas e inactivas alternativamente. La primera máquina en estar activa es B. Supongamos que la máquina A (o B)

está activa. Esta máquina realizará las computaciones oportunas con el valor de entrada y escribirá en su cinta de comunicación enviando un mensaje a B (o A). Es entonces cuando A (o B) pasa a estar inactivo y se activa B (o A), a menos que el protocolo haya terminado. Al final del protocolo, la máquina B decide si acepta o rechaza el valor de entrada escribiendo en la cinta de salida un 1 ó 0 respectivamente.

Para la definición de sistema de pruebas interactivas necesitaremos unas nociones previas.

Nota 2.1. Denotamos $A(x, r; \alpha_1, ..., \alpha_t)$ al mensaje que envía A, cuando tiene entrada x, la cinta aleatoria contiene la secuencia r y ha recibido los mensajes $\alpha_1, ..., \alpha_t$. Podemos abreviarlo si no da lugar a confusión como $A(x)$.

Definición 2.12 ([9])**.** Sea (A, B) un par de máquinas de Turing interactivas y x su valor de entrada. La distribución de salida de B es el valor de salida del par de máquinas que es un espacio de probabilidad. Se denota por $[A(x), B(x)]$. Este espacio de probabilidad está inducido por las secuencias contenidas en las cintas aleatorias de A y B.

Observación 2.1. Sea $x \in L$, decimos que x es suficientemente grande si para algún $h > 0$ se cumple $|x| > h$.

A partir de ahora, la noción de lenguaje se vuelve muy importante porque va a formar parte de las características de los protocolos interactivos que veremos a continuación. Si un protocolo posee estas características, da lugar a un sistema de pruebas interactivas.

Definición 2.13. [9] Un protocolo interactivo (A, B) es completo si para cada $c > 0$ y para cada $x \in L$ suficientemente grande:

$$P([A(x), B(x)] = 1) \geq 1 - \mid x \mid^{-c} \tag{2.1}$$

donde $P([A(x), B(x)] = 1)$ es la probabilidad de que B acepte el valor $x \in L$.

Un protocolo interactivo completo significa que si el valor de entrada está contenido en el lenguaje entonces B lo aceptará casi con total seguridad.

Definición 2.14. [9] Un protocolo interactivo (A, B) es robusto si para cada $c > 0$, para toda máquina de Turing interactiva A^* y para $x \notin L$ suficientemente grande:

$$P([A(x)^*, B(x)] = 0) \geq 1 - \mid x \mid^{-c} \tag{2.2}$$

donde $P([A(x)^*, B(x)] = 0)$ es la probabilidad de que al entrar el valor $x \notin L$, la máquina B no acepte dicho valor.

Un protocolo interactivo robusto significa que si un elemento no está en el lenguaje para cualquier probador A^*, entonces B no lo aceptará casi con total seguridad.

Definición 2.15. [10] Un sistema de pruebas interactivas en L es un protocolo interactivo completo y robusto.

2.3. Conocimiento cero

En esta sección, veremos otra característica que pueden tener los protocolos interactivos, el conocimiento cero. Haremos distinción entre la característica del conocimiento cero de un protocolo interactivo y una prueba de conocimiento cero. Esta última es la que definiremos al final de esta sección.

La novedad que incluye el conocimiento cero respecto a otros protocolos criptográficos es que la información que se cede al verificador **no** es relevante. Todo a lo que el verificador puede acceder durante la prueba solo es útil para que el probador demuestre que conoce la información pero para nada más, es decir, no puede extraerse la información secreta del probador a través de la información que se transfiere durante la prueba.

La información que es cedida al verificador durante una prueba de conocimiento cero no puede diferenciarse de la que genere un simulador ajeno a la prueba. Para formalizar esta noción, entra en juego la indistinguibilidad, pues es así como se denomina a la incapacidad de discernir entre lo que genera un simulador y la información que obtiene el verificador. Además, dentro de la indistinguibilidad nos encontramos con diferentes grados que serán los que darán lugar a diferentes tipos de conocimiento cero.

Formalicemos primero los conceptos de indistinguibilidad.

2.3.1. Indistinguibilidad

Explicado de una manera sencilla, dos familias de variables aleatorias son indistinguibles si tomando una muestra de una de las dos, no puede deducirse a cuál de las dos pertenece. Para obtener la definición de indistinguibilidad se ha consultado [11] pero las definiciones son bastante abruptas y ha tenido que hacerse una labor extra de interpretación, puesto que el concepto de indistinguibilidad tiene una bibliografía escasa y difícil de encontrar.

Sean $\{U(x)\}_{x\in L}$ y $\{V(x)\}_{x\in L}$ dos familias de variables aleatorias con $L \subset \{0,1\}^*$ donde $U(x) \in \{0,1\}^*$ y $V(x) \in \{0,1\}^*$. Sea J una máquina de Turing que actúa como un "juez". Se toma una muestra aleatoria de $U(x)$ ó $V(x)$, se introduce en J y devuelve un 0 si pertenece a $U(x)$ ó un 1 si pertenece a $V(x)$.

Ocurre que si $x \in L$ es lo suficientemente grande, no puede distinguirse si la muestra proviene de $U(x)$ o de $V(x)$ y entonces la salida de J no tiene sentido, es decir, que tanto si devuelve 1 como si devuelve 0 podría estar acertando. Es en estos casos cuando hablamos de la indistinguibilidad entre las dos familias. Dentro de la indistinguibilidad encontramos distintos tipos en función de dos parámetros: el tamaño de la muestra y el tiempo de computación de J.

Los tipos de indistinguibilidad que veremos son igualdad, estadísticamente indistinguibles y computacionalmente indistinguibles.

Para las próximas definiciones, y si no decimos lo contrario, vamos a considerar dos familias de variables aleatorias y la máquina de Turing J como hasta ahora.

Definición 2.16. Sean dos familias de variables aleatorias $\{U(x)\}_{x\in L}$ y $\{V(x)\}_{x\in L}$ y sea J una máquina de Turing sin límite computacional. Estas dos familias son iguales si tomada una muestra de tamaño arbitrario, la salida de J no tiene sentido.

Definición 2.17. Sea $\{U(x)\}_{x\in L}$ una familia de variables aleatorias y sea $U(x)$ una muestra, se dice que la muestra es de tamaño polinomial si el tamaño de la muestra está acotado por un polinomio, Q, que depende únicamente de $|x|$.

Definición 2.18. Sean dos familias de variables aleatorias $\{U(x)\}_{x\in L}$ y $\{V(x)\}_{x\in L}$ y sea J una máquina de Turing sin límite computacional. $\{U(x)\}_{x\in L}$ y $\{V(x)\}_{x\in L}$ son estadísticamente indistinguibles en L si dada una muestra de tamaño polinomial, entonces la salida de J no tiene sentido.

De una manera más explícita: para toda constante $c > 0$, y dado $x \in L$ lo suficientemente grande, las dos familias de variables aleatorias son estadísticamente indistinguibles si:

$$\sum_{\alpha\in\{0,1\}^*} \mid P(U(x) = \alpha) - P(V(x) = \alpha) \mid < \mid x \mid^{-c} \tag{2.1}$$

donde $P(U(x) = \alpha)$ es la probabilidad de que la variable aleatoria $U(x)$ sea α y lo análogo para $P(V(x) = \alpha)$. Por lo tanto, si las dos familias de variables aleatorias son estadísticamente indistinguibles, las probabilidades de que la muestra tomada sea de $U(x)$ o de $V(x)$ son muy cercanas.

Para definir la indistinguibilidad computacional vamos a recurrir a familias de circuitos booleanos, que van a actuar como el juez, en lugar de una máquina de Turing, como hasta ahora. Al igual que en las máquinas de Turing, devuelve un 0 si pertenece a $U(x)$ ó un 1 si pertenece a $V(x)$.

Recordemos que un circuito booleano es un dispositivo teórico que modela un circuito físico, que dado un valor de entrada devuelve una salida booleana. Dentro de este circuito hay funciones booleanas conocidas como puertas lógicas. En el circuito entra un valor, pasa por las puertas lógicas y después el circuito devuelve una salida booleana.

Sea Y una variable aleatoria e y un resultado de la misma, las familias de circuitos booleanos que vamos a utilizar son $C = \{C_y\}_{y\in Y}$ de tamaño polinómico, donde y es el valor de entrada del circuito. Cada circuito booleano C_y tiene una salida booleana y para algún $e > 0$, todo $C_y \in C$ tiene como mucho $|y|^e$ puertas lógicas, con $|y|$ el tamaño del resultado de la variable aleatoria.

Las familias de variables aleatorias de las que se van a tomar muestras aleatorias deben cumplir lo siguiente: Sea $\{U(x)\}_{x\in L}$ una familia de variables aleatorias a la que denotamos U. Para alguna constante $d > 0$, a toda $U(x) \in U$ se le asigna probabilidad positiva solo a las cadenas de longitud $|x|^d$. Lo que quiere decir es que a la hora de tomar una muestra aleatoria de estas variables aleatorias solo se tomarán las de esa longitud concreta.

Entonces, sean $U = \{U(x)\}_{x\in L}$ y $V = \{V(x)\}_{x\in L}$ dos familias de variables aleatorias donde las muestras que se van a tomar son de longitud $|x|^d$ y sea $C = \{C_y\}_y$, con $y \in U(x)$ ó $y \in V(x)$ la familia de circuitos booleanos antes descrita, denotamos $P(U, C, x)$ a la probabilidad de que el circuito booleano nos devuelva un 0, habiendo tenido como entrada una cadena aleatoria $y \in U(x)$, de acuerdo a como hemos definido $U(x)$. Dicho de otro modo, la probabilidad de que el juez, en este caso el circuito C_y, deduzca que y proviene de $U(x)$. Y denotamos como $P(V, C, x)$ a la probabilidad de que el circuito nos devuelva un 1, teniendo como entrada un $y \in V(x)$. Como vemos, el circuito nos debe devolver un 0 si proviene de $U(x)$ y un 1 si proviene de $V(x)$.

Definición 2.19. Sean $\{U(x)\}_{x\in L}$ y $\{V(x)\}_{x\in L}$ dos familias de variables aleatorias definidas como antes y sea C , también definido como antes, dado $x \in L$ suficientemente grande y $c > 0$. Las dos familias de variables aleatorias son computacionalmente indistinguibles si:

$$| P(U, C, x) - P(V, C, x) | < | x |^{-c} \tag{2.2}$$

Por lo tanto, si $\{U(x)\}_{x\in L}$ y $\{V(x)\}_{x\in L}$ son computacionalmente indistinguibles, estas probabilidades van a ser muy cercanas. Lo que quiere decir que $U(x)$ y $V(x)$ son muy parecidas porque el juez podría decir tanto que es de una como que es de otra y estaría acertando. Dicho de otro modo, dos familias de variables aleatorias son computacionalmente indistinguibles si dada una muestra de tamaño polinomial y una familia de circuitos de tiempo polinomial, entonces el veredicto que dé el circuito es sin sentido.

Para que un protocolo sea de conocimiento cero es necesario que la información que se cede al verificador sea simulable, es decir, que exista un simulador que pueda generar esa información siendo ajeno al protocolo. A grosso modo, si puede simularlo una entidad ajena al protocolo, entonces, es de conocimiento cero. Es aquí donde entra la aproximación de familias de variables aleatorias y esta aproximación es realizada por una máquina de Turing probabilística que hace de simulador.

Al igual que describimos tres tipos de indistinguibilidad, distinguimos tres tipos de aproximaciones:

Definición 2.20. Sea $U = \{U(x)\}_{x\in L}$ una familia de variables aleatorias, decimos que U es perfectamente aproximable en L si existe una máquina de Turing probabilística M, con tiempo esperado polinomial, de modo que para todo $x \in L$, $M(x)$ es igual a $U(x)$

Definición 2.21. Sea $U = \{U(x)\}_{x\in L}$ una familia de variables aleatorias, decimos que U es estadísticamente (resp. computacionalmente) aproximable en L si existe una máquina de Turing probabilística M, con tiempo esperado polinomial, de modo que la familia $\{M(x)\}_{x\in L}$ es estadísticamente (resp. computacionalmente) indistinguible en L de $\{U(x)\}_{x\in L}$ respectivamente.

2.3.2. Introducción al conocimiento cero

Cuando ya hemos definido la indistinguibilidad, vamos a definir un protocolo interactivo de conocimiento cero. Para ello, vamos a adaptar lo visto en la anterior subsección a lo que sabemos de un protocolo interactivo.

Sea un protocolo interactivo (A, B) como en la Definición 2.11, llamamos vista de B a todo lo que observa B durante todo el protocolo. Sean σ y ρ las secuencias de bits de las cintas aleatoria de A y B, y sean a_i y b_i, con $i = \{1, ..., n\}$, los mensajes i-ésimos enviados por A y B respectivamente durante los n turnos. Entonces $(\rho, b_1, a_1, ..., b_n, a_n)$ es la vista de B e inducida por la aleatoriedad de ρ tenemos la variable aleatoria $\langle A, B\rangle(x)$ cuyo valor es $(\rho, b_1, a_1, ..., b_n, a_n)$ cuando tiene de valor de entrada x. Dicho de otro modo, $\langle A, B\rangle(x)$ es la variable aleatoria que devuelve la vista de B durante el protocolo, cuando tiene de valor de entrada x. Por ello, denotamos $\{\langle A, B\rangle(x)\}_{x\in L}$ a la familia de variables aleatorias generadas para todo $x \in L$.

Teniendo en cuenta lo visto en las Definiciones 2.20 y 2.21, vamos a definir el conocimiento cero en un protocolo interactivo.

Definición 2.22. [11] Sea (A, B) un protocolo interactivo en $L \subset \{0,1\}^*$ y sea B^* una máquina de Turing interactiva cualquiera. Decimos que (A, B) es un protocolo interactivo de conocimiento cero perfecto (resp. estadístico, computacional), si la familia de variables aleatorias $\{\langle A, B^*\rangle(x)\}_{x\in L}$ es perfectamente (resp. estadísticamente, computacionalmente) aproximable en L.

Vemos que lo que observa el verificador durante el protocolo y lo que genera una máquina de Turing probabilística será indistinguible perfecta, estadística o computacionalmente. Por lo tanto, un protocolo interactivo puede ser de conocimiento cero perfecto, estadístico o computacional.

Una vez visto esto, pasemos a las pruebas de conocimiento cero.

2.4. Pruebas de conocimiento cero

En esta sección tratamos finalmente la definición formal de una prueba de conocimiento cero, primero dando la versión más práctica y posteriormente la formal.

Recordemos que una prueba de conocimiento cero es un protocolo criptográfico con dos entidades: el probador y el verificador, donde el probador debe demostrar al verificador que conoce una información secreta sin revelarla. [12] Las principales características de las pruebas de conocimiento cero son:

1.- El verificador no puede obtener nada útil del protocolo, es decir, no puede obtener información sensible del protocolo.

2.- El verificador no tiene capacidad de hacerse pasar por el probador que ha verificado, es decir, los mensajes que se intercambian en el protocolo no pueden utilizarse para hacerse pasar por el probador que se está verificando.

3.- La probabilidad de que el verificador acepte una declaración falsa como verdadera es muy pequeña.

4.- La probabilidad de que el verificador sea convencido de una declaración que es verdadera es muy alta.

Las pruebas de conocimiento cero son protocolos interactivos robustos (Definición 2.14), completos (Definición 2.13) y de conocimiento cero (Definición 2.22).También puede pensarse como un sistema de pruebas interactivas (Definición 2.15) de conocimiento cero.

Definición 2.23. [11] Sea (A, B) un sistema de pruebas interactivas en $L \subset \{0,1\}^*$, (A, B) es de conocimiento cero perfecto (resp. estadístico, computacional) si es un protocolo interactivo de conocimiento cero perfecto (resp. estadístico, computacional).

Si observamos las características que tienen las pruebas de conocimiento cero, vemos que un sistema de pruebas de conocimiento cero también las cumplirá: Las características 1 y 2, las cumple debido a la propiedad de conocimiento cero del sistema de pruebas; la característica 3 es la robustez por ser un sistema de pruebas interactivas; y la 4 es la completitud, también de un sistema de pruebas interactivas.

2.5. Lenguajes en los protocolos.

Hasta ahora, las pruebas de conocimiento cero que hemos definido están basadas en demostrar la pertenencia de un elemento a un lenguaje. Esta idea, será recogida por otros autores [8] para obtener otros tipos de pruebas de conocimiento cero que veremos en la siguiente sección y en el Capítulo 4.

La seguridad de este tipo de pruebas reside en que demostrar que un elemento pertenece a un lenguaje es un problema de decisión. En esta sección daremos algunas nociones sobre estos problemas y la relación que tienen con los lenguajes que se utilizan en las pruebas de conocimiento cero.

Definición 2.24. Un problema es P si existe un algoritmo que puede resolverlo en tiempo polinómico. Si no puede resolverse en este tiempo, se dice que es NP.

Definición 2.25. Un lenguaje es P si demostrar que un elemento pertenece a ese lenguaje es un problema P. Lo análogo para los lenguajes NP.

Podemos ver que si un lenguaje es P, el verificador podrá probar que el elemento está en el lenguaje sin necesidad de interactuar con el probador. Por otro lado, si el lenguaje es NP al verificador le resultará más complicado probarlo. Es por esto que los lenguajes a los que se recurrirá para las pruebas de conocimiento cero serán NP.

En el siguiente capítulo explicaremos pruebas de conocimiento cero sobre distintos lenguajes. Los autores [11] [9] recurren a una nueva definición de lenguajes:

Definición 2.26. Un lenguaje es un conjunto para el cual es computacionalmente difícil probar la membresía de un elemento.

Un lenguaje es computacionalmente difícil porque es un lenguaje P ó NP. Aunque, los lenguajes que veremos serán NP.

Los lenguajes que veremos en el siguiente capítulo son los isomorfismos de grafos, los grafos 3-coloreables y los residuos cuadráticos. Estos lenguajes son válidos para todas las definiciones vistas en este capítulo.

2.6. Más allá de los lenguajes.

Las pruebas de conocimiento cero demuestran el conocimiento de una información sin revelarla. Hemos visto que prueban la pertenencia a un lenguaje en las primeras menciones del conocimiento cero [9–11].

Esta idea evoluciona con Fiat y Shamir en [8] para dar solución a la identificación sin uso de contraseñas. Si un tercero intercepta la contraseña de un individuo, éste podría hacerse pasar por él. El concepto básico de las pruebas de conocimiento cero para la identificación es el siguiente: el probador, en conocimiento de un secreto que lo identifica, demuestra su identidad al verificador sin transmitirle ese secreto.

Los conceptos que hemos tratado en este capítulo como la robustez, la completitud y la indistinguibilidad se basan en la pertenencia al lenguaje y se adaptarán a la identificación [8].

La completitud y la robustez se basaban en la pertenencia o no de un elemento a un lenguaje, en la identificación se hace referencia al conocimiento o no de un secreto.

Definición 2.27. Sea (A, B) un protocolo interactivo, (A, B) es completo si la probabilidad de que B acepte la identidad de A cuando A posee el secreto, es muy cercana a 1.

Definición 2.28. Sea (A, B) un protocolo interactivo, (A, B) es robusto si la probabilidad de que B acepte la identidad de A cuando A no posee el secreto, es muy cercana a 0.

El concepto de indistinguibilidad se mantiene, es decir, se debe comprobar que la vista del verificador durante el protocolo es susceptible de ser generada por un simulador ajeno a la prueba. El matiz distinto es que las familias de variables aleatorias no se forman por los diferentes elementos del lenguaje, si no por otras variables que veremos en el Capítulo 4.

Como vemos, la idea de no transmitir conocimiento se mantiene y estos autores la utilizan para crear algoritmos de identificación y de firma que veremos en los Capítulos 4 y 5.

Capítulo 3

Pruebas de conocimiento cero sobre lenguajes

Después de haber definido las pruebas de conocimiento cero sobre lenguajes en el capítulo anterior, en este capítulo vamos a explicar tres tipos de pruebas de conocimiento cero sobre lenguajes. Recordemos que en las pruebas de conocimiento cero sobre lenguajes, el probador demuestra al verificador que un elemento pertenece a un lenguaje sin dar más información que la pertenencia.

En el capítulo anterior habíamos definido los lenguajes como subconjuntos de $\{0,1\}^*$, ahora, como adelantábamos en la Sección 2.5, se redefinen los lenguajes como conjuntos para los que demostrar que un elemento pertenece a ellos es computacionalmente difícil o, dicho de otra forma, lenguajes NP. Estos lenguajes mantienen todas las definiciones vistas hasta ahora [11].

Los lenguajes sobre los que vamos a construir pruebas de conocimiento cero son: los isomorfismos de grafos, los grafos 3-coloreables y los residuos cuadráticos. Como explicamos en la Sección 2.5, los lenguajes que vamos a utilizar son conjuntos para los que demostrar la pertenencia es un problema NP.

A continuación, explicamos los tres protocolos para pruebas de conocimiento cero sobre lenguajes: primero, presentamos la base matemática, luego, el problema de pertenencia, y finalmente, el protocolo. También realizaremos un desarrollo más exhaustivo de la prueba de conocimiento cero sobre los grafos 3-coloreables utilizando las definiciones del capítulo anterior.

Nota 3.1. En los próximos apartados utilizaremos la notación $A \to B$ para indicar el paso en el que A realiza las computaciones oportunas y envía el resultado a B. Lo análogo para $B \to A$.

3.1. Isomorfismo de grafos

En primer lugar, expondremos algunas nociones sobre grafos que vamos a utilizar en esta y en la siguiente sección.

Sea C un conjunto, denotamos por $\sigma(C)$ al conjunto de todas las permutaciones del conjunto C. Si tomamos $c \in_R C$ estamos escogiendo un elemento de C aleatoriamente y con una distribución de probabilidad uniforme.

Denotamos $G(V, E)$ al grafo no dirigido con V el conjunto de sus vértices y E el conjunto de sus aristas. Llamamos n al tamaño de V y m al tamaño de E.

Definición 3.1. Sean $G(V, E)$ y $H(V, F)$ dos grafos no dirigidos, son isomorfos si y solo si existe una permutación $\pi \in \sigma(V)$ de modo que $(u, v) \in E \Leftrightarrow (\pi(u), \pi(v)) \in F$. Dicho de otro modo, si dos vértices son adyacentes en un grafo, las imágenes por π lo serán en el otro grafo.

El conjunto de los pares de grafos isomorfos es un lenguaje que denotamos por GI, donde $GI = \{(G, H) : \exists \pi$ de modo que $H = \pi G\}$. Además, decimos que un grafo $H(V, F)$ es una copia isomorfa aleatoria del grafo $G(V, E)$ si H se obtiene de G mediante $\pi \in_R \sigma(V)$ y tomando $F = \{(\pi(u), \pi(v)) : (u, v) \in E\}$.

El problema del isomorfismo de grafos es el siguiente: dados dos grafos hay que determinar si son isomorfos (que pertenecen a GI).

A continuación, presentamos la prueba de conocimiento cero sobre el lenguaje GI

Protocolo 3.1. Este protocolo puede consultarse en [9].

$G_0(V, E_0)$ y $G_1(V, E_1)$ son dos grafos no dirigidos que tanto el probador, A, y el verificador, B, conocen. ϕ es un isomorfismo entre G_0 y G_1, es decir, $G_1 = \phi G_0$ que conoce A.

a) *Fase previa.* A debe demostrar a B que hay un isomorfismo entre G_0 y G_1 sin revelar ϕ.

b) *Intercambio de mensajes.* Los siguientes pasos se repiten n veces, con n el número de vértices de ambos grafos, y en todas las repeticiones, las elecciones aleatorias son nuevas.

   1.- $A \to B$: El probador toma $\pi \in_R \sigma(V)$ y construye una copia aleatoria isomorfa de G_1 a la que denotamos $H(V, F)$. El probador, A, envía al verificador, B, el grafo $H(V, F)$.

   2.- $B \to A$: El verificador, B, toma $e \in_R \{0, 1\}$ y se lo envía a A.

   3.- $A \to B$: El probador comprueba primero si $e \in \{0, 1\}$, si no fuera así pararía. Si $e = 1$, entonces el probador, A, envía π a B; y si $e = 0$, A, envía $\pi\phi$ a B.

   4.- Una vez recibe B la permutación, comprueba si es un isomorfismo entre G_e y H. Si lo es, continua con la siguiente iteración; si no, rechaza y para.

c) *Finalización del protocolo.* Cuando se completan las n iteraciones con éxito, el verificador, B, acepta y para.

Observación 3.1. El problema del isomorfismo de grafos se demostró que era un problema que se encontraba entre P y NP en 2016, fue demostrado por Babai en [1]. Por lo tanto, sería computacionalmente más fácil romper este protocolo porque se puede encontrar un ismorfismo entre dos grafos.

3.2. Grafos 3-coloreables

Definición 3.2. Sea $G(V, E)$ un grafo no dirigido, es un grafo 3-coloreable si existe una aplicación $\phi : V \to \{1, 2, 3\}$ de tal manera que a todo par de vértices adyacentes se le asignan colores distintos, es decir, $\forall (u, v) \in E, \phi(u) \neq \phi(v)$.

El conjunto de grafos no dirigidos que son 3-coloreables es un lenguaje que denotamos $G3C$.

El problema de los grafos 3-coloreables es el siguiente: dado un grafo hay que determinar si es 3-coloreable. El problema es NP por lo que el lenguaje $G3C$ es NP [9].

Observación 3.2. El interés del lenguaje 3-coloreable es debido a que demostrar que un grafo es 3-coloreable es un problema NP, algo que no ocurre con los grafos 4-coloreables. Para los grafos 4-coloreables asociados a un mapa existe un teorema que demuestra que todo grafo puede ser coloreado por cuatro colores sin que los vértices adyacentes tengan el mismo color.

Ahora, presentamos una prueba de conocimiento cero sobre el lenguaje $G3C$.

Protocolo 3.2. Este protocolo puede consultarse en [9].

a) *Fase previa.*

$G(V, E)$ es un grafo no dirigido que A y B conocen, con V de tamaño n y E de tamaño m. ϕ es una coloración correcta que solo conoce A. A debe demostrar a B que el grafo G es 3-coloreable (que pertenece a $G3C$) sin revelar ϕ.

b) *Intercambio de mensajes.*

Los siguientes pasos se repiten m^2 veces y cada una de las veces las elecciones aleatorias son nuevas:

1.- $A \to B$: A toma $\pi \in_R \sigma(\{1, 2, 3\})$, calcula $\pi(\phi(v))$ $\forall v \in V$. Cada uno de los $\pi(\phi(v))$ son encriptados[1] individualmente por A y enviados a B.

2.- $B \to A$: B escoge una arista $e = (u, v) \in E$ y se la envía a A.

3.- $A \to B$: A comprueba que $e \in E$ y envía a B las desencriptaciones de u y v.

4.- B recibe la desencriptación, desencripta u y v y comprueba que contienen diferentes elementos de $\{1, 2, 3\}$. Si los elementos de $\{1, 2, 3\}$ son distintos, continúa con la siguiente iteración. Si no puede desencriptar o los números que contienen son iguales, entonces B rechaza y para.

De esta forma, el verificador no conocerá la coloración real del grafo porque la coloración está oculta mediante π

c) *Final del protocolo.*

Si se completan las m^2 iteraciones con éxito, entonces B acepta y para.

[1]Se utiliza cualquier algoritmo de encriptación que permita encriptarlos y desencriptarlos de manera individual

Proposición 3.1. El Protocolo 3.2 es una prueba de conocimiento cero computacional para $G3C$.

Demostración. Tenemos en cuenta que es un protocolo interactivo porque las entidades que participan en el protocolo pueden formalizarse como máquinas de Turing interactivas probabilísticas, siendo B una máquina de Turing de tiempo esperado polinomial y A sin límite computacional. Estas entidades interactuarán como lo hacen las máquinas de Turing en los protocolos interactivos.

Como una prueba de conocimiento cero computacional es un sistema de pruebas de conocimiento cero computacional vamos a probar que es completo, robusto y de conocimiento cero computacional. Utilizando la Definición 2.23:

- *Completitud.* Basándonos en la Definición 2.13, si el grafo de entrada $G \in G3C$ entonces, cualquier par de vértices adyacentes que se tomen van a tener color distinto, por lo que la probabilidad de que el verificador acepte dicho grafo es 1. El Protocolo 3.2 es completo.

- *Robustez.* Utilizando la Definición 2.14, supongamos ahora que el grafo de entrada $G \notin G3C$ lo que quiere decir que al menos habrá un par de vértices adyacentes que no tendrán la coloración adecuada. La probabilidad de que cualquier verificador rechace este grafo es al menos $1/m$ que es la probabilidad de escoger la arista que une los vértices del mismo color. Por lo tanto, la probabilidad de que el verificador acepte el grafo es como mucho $(m-1)/m$ en cada iteración, que es la probabilidad de escoger un par de vértices bien coloreados. En total en todo el protocolo la probabilidad de que acepte un $G \notin G3C$ es como mucho $((m-1)/m)^{m^2}$, una probabilidad ínfima y por consiguiente, el Protocolo 3.2 es robusto.

- *Conocimiento cero computacional.* Para esta demostración utilizaremos las Definiciones 2.19 y 2.21. La información que obtiene el verificador en cada paso es $\pi(\phi(u))$ y $\pi(\phi(v))$ donde (u, v) son vértices adyacentes escogidos aleatoriamente por el verificador, y donde π también es escogido aleatoriamente por el probador. Esta información es lo que conocemos como la vista del verificador durante el protocolo y es una variable aleatoria inducida por la aleatoriedad de la elección de π. Como cada $G \in G3C$ da lugar a una variable aleatoria, construimos la familia de variables aleatorias que denotamos por $\{\langle A, B\rangle(G)\}_{G \in G3C}$.

  Basándonos en la Definición 2.21 consideremos como simulador una máquina de Turing M probabilística que con entrada un grafo $G \in G3C$ nos devuelve dos enteros $(i, j) \in_R \{1, 2, 3\}$ con $i \neq j$. El conjunto de las salidas de esta máquina de Turing es una familia de variables aleatorias que denotaremos por $\{M(G)\}_{G \in G3C}$.

  Probemos que estas dos familias de variables aleatorias son computacionalmente indistinguibles (Definición 2.19). Primero, necesitamos construir al "juez", que en el caso de la indistinguibilidad computacional es un circuito booleano:

  Sea $C = \{C_y\}$ nuestro juez con $y \in \langle A, B\rangle(G)$ ó $y \in M(G)$, y sean las muestras y todas del mismo tamaño por definición. Tenemos que $P(\{\langle A, B\rangle(G)\}, C, G)$ es la probabilidad de que el circuito devuelva un 1 al introducir un $y \in \langle A, B\rangle(G)$ y $P(\{M(G)\}, C, G)$ la probabilidad de que devuelva un 0 al introducir un $y \in M(G)$.

Como vemos, los $y \in \langle A, B\rangle$ y los $y \in M(G)$ no guardan una relación con su procedencia porque los $y \in \{(1,2),(2,1),(1,3),(3,1),(2,3),(3,2)\}$ sea cual sea su procedencia. Por lo que si calculamos $|P(\{\langle A, B\rangle(G)\}, C, G) - P(\{M(G)\}, C, G)|$(2.2) obtendremos un valor muy cercano a 0 porque las probabilidades de que acierte o falle son las mismas en cada caso. Por lo tanto, podemos afirmar que estas dos familias de variables aleatorias son computacionalmente indistinguibles, lo que convierte al Protocolo 3.2 en un sistema de pruebas interactivas de conocimiento cero computacional.

□

3.3. Residuos cuadráticos

Vamos a fijar algunas nociones sobre residuos cuadráticos tomadas de [11] que utilizaremos tanto para definir el lenguaje como para presentar el problema de los residuos.

Definición 3.3. Sea $p \in \mathbb{N}$ y sea $\mathbb{Z}_p^*$ el conjunto de los invertibles módulo p. Entonces, un $q \in \mathbb{Z}_p^*$ es un residuo cuadrático módp si existe un $w \in \mathbb{Z}_p^*$ de modo que $w^2 = q \bmod p$. Puede determinarse en tiempo polinomial que $q \in \mathbb{Z}_p^*$.

Además $q \in \mathbb{Z}_p^*$ es un residuo cuadrático mód p si y solo si q es un residuo cuadrático módulo todos los factores primos de p.

Definición 3.4. Denotamos por $Q_p(q)$ al operador:

$$Q_p(q) = \begin{cases} 0 \text{ si } q \text{ es un residuo cuadrático mód} p \\ 1 \text{ en otro caso} \end{cases} \tag{3.1}$$

Dados $p \in \mathbb{N}$, $q \in \mathbb{Z}_p^*$ y la factorización de p, el operador $Q_p(q)$ puede calcularse en tiempo polinomial.

Definición 3.5. Sea $q \in \mathbb{Z}_p^*$ y sea $\prod_{i=1}^{k} p_i^{\alpha_i}$ la descomposición en primos de p, entonces el símbolo de Jacobi de q mód p se define como:

$$\left(\frac{q}{p}\right) = \prod_{i=1}^{k} \left(\frac{q}{p_i}\right)^{\alpha_i} \tag{3.2}$$

donde $\left(\frac{q}{p_i}\right) = +1$ si q es un residuo cuadrático módp_i y $\left(\frac{q}{p_i}\right) = -1$ en otro caso. Recordemos que estos $\left(\frac{q}{p_i}\right)$ para $i = 1, ..., k$ son los símbolos de Legendre.

Es importante destacar, que si un elemento tiene símbolo de Jacobi +1 no implica que sea un residuo cuadrático pero si al revés.

Basándonos en esta definición, el símbolo de Jacobi puede calcularse en tiempo polinomial.

El lenguaje que utilizaremos es QR (del inglés *quadratic resiudosity*) y lo definimos de la siguiente forma:

$$QR = \{(p, q) : p \in \mathbb{N}, q \in \mathbb{Z}_p^*, Q_p(q) = 0\} \tag{3.3}$$

con p y q expresados en binario.

El problema de la residualidad cuadrática es calcular $Q_p(q)$ dados $p \in \mathbb{N}$, $q \in \mathbb{Z}_p^*$ y $\left(\frac{p}{q}\right) = +1$, es decir, demostrar que $(p, q) \in QR$.

El siguiente protocolo basará su funcionamiento en las siguientes características de los residuos cuadráticos: sea $p \in \mathbb{N}$, $q, u \in \mathbb{Z}_p^*$, entonces:

- Si $Q_p(q) = Q_p(u) \Rightarrow Q_p(qu) = 0$.
- Si $Q_p(q) \neq Q_p(u) \Rightarrow Q_p(qu) = 1$.

Todas las nociones sobre residuos cuadráticos vistas en esta sección serán necesarias para próximas secciones.

Protocolo 3.3. Este protocolo puede consultarse en [11]. A quiere demostrar a B que $(p, q) \in QR$ sin mostrarle la evidencia por la que q es un residuo cuadrático módulo p.

a) *Fase previa.*

A y B conocen (p, q) y A conoce la evidencia de que q es un residuo cuadrático módulo p.

b) *Intercambio de mensajes.* Los siguientes pasos se repiten m veces y las elecciones aleatorias se hacen de nuevo en cada paso. m es el número de dígitos de p en binario.

1.- $A \to B$: A escoge aleatoriamente un residuo cuadrático módp que denotamos u y se lo envía a B.

2.- $B \to A$: B escoge $e \in_R \{0, 1\}$ y se lo envía a A.

3.- $A \to B$: A escoge aleatoriamente $+$ ó $-$ de $v = \pm\sqrt{uq^e \text{ mód } p}$ y se lo envía a B.

B recibe v, calcula v^2 y comprueba que $v^2 = uq^e$. Si no fuera así rechazaría y pararía.

c) *Final del protocolo.*

Si se realizan las m repeticiones con éxito, entonces B acepta y para.

Capítulo 4

Pruebas de conocimiento cero para la identificación

En la Sección 2.6, al final de Capítulo 2, dimos un adelanto de lo que serían las pruebas de conocimiento cero adaptadas a la identificación.

En una prueba de conocimiento cero para la identificación, el probador, en posesión de un secreto que lo identifica, demuestra al verificador su identidad, sin desvelarle el secreto.

En este capítulo profundizaremos más en esta variante de pruebas de conocimiento cero. Realizaremos un desarrollo más completo de dos de los protocolos.

4.1. De la membresía a la identificación

Fiat y Shamir [8] toman la idea de la identificación a través de pruebas de conocimiento cero y la combinan con las aportaciones de Goldwasser [10] (Capítulos 2 y 3) y el planteamiento de Shamir [17].

A continuación, expondremos algunas ideas de [17] que nos serán útiles para describir las pruebas de conocimiento cero para la identificación.

Shamir propuso un protocolo criptográfico para que un grupo de usuarios se comunicasen entre ellos de forma segura sin la necesidad de intercambiar claves públicas o privadas y sin recurrir a terceros. Aparecían en este protocolo cadenas de valores que contenían la información necesaria para que el usuario pudiera encriptar y firmar los mensajes que enviaba, y, desencriptar y verificar los mensajes que recibía. Estas cadenas se denominaban tarjetas electrónicas y cada usuario poseía una. Además, no necesitaban actualizarse cuando ya habían sido emitidas y entraba un nuevo usuario.

En el protocolo de [17] aparecía una nueva entidad: el centro de confianza, que es el que emitía las tarjetas electrónicas a los usuarios cuando se unen por primera vez al protocolo. Los centros de confianza desaparecen cuando se han emitido todas las tarjetas correspondientes haciendo que la actividad no dependa del centro.

La utilidad del centro de confianza en el protocolo de Shamir [17] es la siguiente: el usuario se identifica mediante métodos no criptográficos que dependen del contexto al centro de confianza (en [17] se escoge su nombre y correo electrónico como su clave pública), de manera que pueda identificarse, no retractarse y que para otro usuario esa identificación sea válida. Cuando el usuario ya está identificado, el centro de confianza

adjudica la cadena de información (la tarjeta electrónica) al usuario. El centro de confianza estará en posesión de algún tipo de información que le permitirá calcular las claves privadas de todos los usuarios. Que el centro de confianza tenga acceso a las claves no es peligroso puesto que estas claves sirven tan solo para la identificación y cada usuario ya se ha identificado al centro de confianza antes. La diferencia con un sistema de clave pública es que en él se generan la clave pública y privada aleatoriamente, mientras que en este protocolo la genera el centro de confianza después de que el usuario se identifique.

El protocolo de Shamir [17] puede aplicarse a grupos cerrados de usuarios donde hay una entidad en la que todos confían.

Una vez vistas las nociones que presenta Shamir, pasemos a la identificación con pruebas de conocimiento cero.

Nota 4.1. En las próximas secciones denominaremos *tarjeta electrónica* ó simplemente *tarjeta* a la cadena de información que cada usuario posee y que contiene los valores para encriptar, desencriptar, firmar y verificar los mensajes que envía y recibe.

Nota 4.2. Seguiremos utilizando la misma notación que hasta ahora: A es el probador del protocolo y B, el verificador.

Nota 4.3. En los próximos protocolos se mantiene la idea de la identificación: el probador, A, quiere demostrar al verificador, B, que es él.

4.2. Protocolos de Fiat y Shamir

Fiat y Shamir [8] presentan una combinación de los planteamientos de Shamir [17], en cuanto al centro de confianza, el nuevo método de elegir clave pública y tarjetas electrónicas, y la idea de prueba de conocimiento cero de Goldwasser [10].

Fruto de esta combinación se obtiene el protocolo de identificación que explicaremos a continuación y uno de los protocolos de firma digital que explicaremos en el capítulo siguiente.

El siguiente protocolo que se muestra en esta sección posee las siguientes características:

- El centro de confianza de este protocolo es como el de [17]: después de comprobar la identidad de los usuarios, les adjudica las tarjetas electrónicas y no entra más en la dinámica del protocolo.
- El número de usuarios a los que pueden adjudicarse tarjetas es ilimitado.
- No es necesario que exista una lista con todos los usuarios.
- Los verificadores no pueden reproducir o copiar las tarjetas con las que interactúan y tampoco pueden crear nuevos usuarios o modificar identidades ya existentes si conocen de forma parcial o total el contenido de las tarjetas.

En los próximos protocolos nos encontraremos con dos nuevas variables: k y t. t denota el número de iteraciones que se realizan y k el número de elementos que se escogen e intercambian.

Antes de presentar el protocolo vamos a definir un tipo de funciones que se utilizan en éste y otros protocolos.

Definición 4.1. Una función DES es una función de encriptado de bloque de clave simétrica que toma cadenas de longitud fija y la transforma en otra cadena de longitud fija. Obtener la imagen es computacionalemente fácil, mientras que obtener la antiimagen es computacionalmente difícil [18].

Protocolo 4.1 (**Fiat Shamir** [8])**.** El probador, A, quiere demostrar su identidad al verificador, B.

a) *Preparación.* Se fijan dos enteros k y t. El centro de confianza realiza los siguientes pasos:

- Escoge dos números primos secretos p y q, y publica su producto $n = pq$.
- Publica una función DES, f, que asigna valores en $\mathbb{Z}_n$.

El probador se identifica al centro de confianza mediante métodos no criptográficos. Después, el centro crea una cadena I que contiene información sobre el usuario(nombre, dirección, correo) y sobre la tarjeta(fecha de expiración, limitaciones de validez). Para crear esta tarjeta realiza los siguientes pasos:

- Calcula un número indeterminado de $v_j = f(I, j)$ para $j = 0, 1, ...$
- De todos los v_j escoge k cualesquiera de modo que v_j es un residuo cuadrático módulo n. Para simplificar la notación tomaremos $j = 1, ..., k$.
- Calcula s_j que es la raíz cuadrada más pequeña de v_j^{-1} mód n para $j = 1, ..., k$.
- El centro emite como tarjeta para A la cadena $(I; s_1, ..., s_k)$. Los s_j son el secreto de A.

b) *Intercambio de mensajes* Estos pasos se realizan t veces:

1.- $A \to B$: A envía I a B.

2.- B genera los $v_j = f(I, j)$ para $j = 1, ..., k$.

Los pasos del 3 al 6 se repiten para $i = 1, ..., t$:

3.- $A \to B$: A escoge un r_i de modo que $1 \leq r_i < n$ y envía a B el valor $x_i = r_i^2$ mód n.

4.- $B \to A$: B escoge un vector binario aleatorio $(e_{i1}, ..., e_{ik})$ y se lo envía a A.

5.- $A \to B$: A calcula $y_i = r_i \cdot \prod_{e_{ij}=1} s_j$ mód n y envía y_i a B.

6.- B comprueba que $x_i = y_i^2 \cdot \prod_{e_{ij}=1} v_j$ mód n. Si se cumple la igualdad, pasa a la siguiente iteración si no, rechaza y para.

c) *Final del protocolo.* Si se completan las t iteraciones con éxito, entonces el verificador acepta y para.

Observación 4.1. Algunas observaciones sobre el Protocolo 4.1:

- Se escoge $kt = 20$ por razones de seguridad [8] para que el protocolo sea robusto.
- El verificador aunque posea los v_j no puede obtener los s_j porque no conoce la descomposición de n para $j = 1, ..., k$.

Vamos a probar que este protocolo es una prueba de conocimiento cero computacional. Para ello, utilizaremos las nociones dadas en el Capítulo 2 y, más en concreto, las de la Sección 2.6. Tendremos en cuenta todas las notaciones y valores obtenidos del Protocolo 4.1.

Proposición 4.1. El Protocolo 4.1 es una prueba de conocimiento cero computacional [8].

Demostración. Para demostrar que es una prueba de conocimiento cero, vamos a probar que es un protocolo interactivo completo, robusto y de conocimiento cero computacional.

- *Completitud.*

  Vamos a suponer que A conoce los s_j para $j = 1, ..., k$. Si A sigue el protocolo con normalidad, entonces A calcula $y_i = r_i \cdot \prod_{e_{ij}=1} s_j \text{ mód } n$ y se lo envía a B. La comprobación que hace B es ver si x_i es igual que $y_i^2 \cdot \prod_{e_{ij}=1} v_j \text{ mód } n$. Veamos si se cumple:

  $$y_i^2 = r_i^2 \cdot \prod_{e_{ij}=1} s_j^2 \text{ mód } n =_{(1)} r_i^2 \cdot \prod_{e_{ij}=1} (1/v_j) \text{ mód } n \tag{4.1}$$

  (1) es por la definición de los s_i.

  $$\Rightarrow y_i^2 \cdot \prod_{e_{ij}=1} v_j \text{ mód } n =_{(2)} r_i^2 \cdot \prod_{e_{ij}=1} (1/v_j) \cdot \prod_{e_{ij}=1} v_j \text{ mód } n = \tag{4.2}$$

  (2) es debido a (4.1).

  $$= r_i^2 \cdot \prod_{e_{ij}=1} (1/v_j) v_j \text{ mód } n = r_i^2 \text{ mód } n = x_i \text{ mód } n \tag{4.3}$$

  Al obtener la última igualdad, válida para $i = 1, ..., t$, B aceptará en todas las iteraciones y, por consiguiente, aceptará la identidad de A con probabilidad 1. Por lo tanto, el Protocolo 4.1 es completo.

- *Robustez.* Supongamos que una entidad cualquiera, A^*, es un impostor que no conoce los s_j con $j = 1, .., k$ y quiere hacerse pasar por A. Puede escoger los e_{ij} que vaya a coger B en el paso 4 (Con muchísima suerte, porque tendría que predecir un vector aleatorio que vaya a sacar B en el siguiente paso). Escogería, también, un y_i aleatorio de forma que $x_i = y_i^2 \prod_{e_{ij}=1} v_j$ y $r_i = y_i$ con los e_{ij} que luego escoja B. Así, podría darse que el verificador B aceptara la identidad de A^* con una probabilidad 2^{-k} en cada iteración, que es la probabilidad de que A escoja el mismo vector e_{ij} que tome B en el siguiente paso y una probabilidad total en el protocolo de 2^{-kt}. Por la tanto, si tomamos $kt = 20$, que es como se implementa habitualmente este protocolo por motivos de seguridad (Observación 4.1), la probabilidad de que el verificador acepte cuando el probador es un impostor es de 2^{-20}, una probabilidad ínfima. Así que el Protocolo 4.1 es robusto.

- *Conocimiento cero computacional.*

  La demostración que seguían Fiat y Shamir en [8] resultaba muy tediosa, por lo que opté por una demostración alternativa utilizando la misma idea que en la demostración del Protocolo 3.1.

  En esta parte de la demostración haremos un pequeño cambio para aligerar la notación: todas las variables que tengan el subíndice i para $i = 1, ..., t$, lo "pierden". Esto es debido a que la iteración en la que se encuentre no es importante para el desarrollo de la demostración.

  Para probar que es una prueba de conocimiento cero computacional necesitamos probar que existe un simulador que puede generar la vista del verificador durante el protocolo.

  La información que obtiene el verificador durante el protocolo es el par (x, y) donde x es el cuadrado de r, un entero escogido por el probador, y donde y se obtiene de la elección del vector de bits aleatorios del verificador y de r. Esta información la conocemos como vista de B, es una variable aleatoria que denotamos $\langle A, B\rangle(r)$ y su resultado es un (x, y). Inducida por el entero r, tenemos una familia de variables aleatorias que denotaremos por $\{\langle A, B\rangle(r)\}_{r\in(0,n)}$.

  Como simulador vamos a tomar una máquina de Turing, M, que dados k residuos cuadráticos módulo n, $v = (v_1, ..., v_k)$, tiene como salida el par (x, y), donde y se escoge aleatoriamente y $x = y^2 \cdot \prod_{\bar{e_j}=1} v_j$, con los $\bar{e_j}$ los bits escogidos aleatoriamente por M, $j = 1, ..., k$. Aunque la manera de obtener el par (x, y) no es la misma que en la vista del verificador, están definidos de un modo similar. Las salidas de $M(v)$ forman una variable aleatoria inducida por la aleatoriedad de y y de los $\bar{e_j}$. Y denotamos $\{M(v)\}_v$ a la familia de variables aleatorias inducidas por el vector v.

  Para probar que $\{\langle A, B\rangle(r)\}_{r\in(0,n)}$ y $\{M(v)\}_v$ son computacionalmente indistinguibles recurriremos a los circuitos booleanos.

  Como juez, escogemos a la familia de circuitos booleanos $C = \{C_z\}$ en la que $z = (x, y) \in \langle A, B\rangle(r)$ ó $z = (x, y) \in M(v)$. Si escogemos el circuito C_z, éste debe devolver un 1 si $z \in \langle A, B\rangle(r)$ y un 0 si $z \in M(v)$, además, por definición todos los z son del mismo tamaño. Denotamos por $P(\{\langle A, B\rangle(r)\}, C, z)$ a la probabilidad de que el circuito devuelva un 1 si tiene como entrada un $z \in \langle A, B\rangle(r)$ y por $P(\{M(v)\}, C, z)$ a la probabilidad de que el circuito devuelva un 0 si tiene como entrada un $z \in M(v)$.

  Por cómo hemos construido el simulador, los z que se toman tanto de $\langle A, B\rangle(r)$ como de $M(v)$ no van a guardar relación con su origen, es decir, el "juez", en este caso el circuito C_z, devolverá un 1 ó un 0 indistintamente, pues un $z \in \langle A, B\rangle(r)$ y un $z \in M(v)$ no van a poder diferenciarse. Por lo tanto, basándonos en la fórmula (2.2) de la Definición 2.19 la diferencia $|P(\{\langle A, B\rangle(r)\}, C, z) - P(\{M(v)\}, C, z)|$ es muy cercana a 0 porque la probabilidad de que el juez dictamine que el z proviene de uno o de otro es prácticamente la misma . En consecuencia, las familias $\{\langle A, B\rangle(r)\}_{r\in(0,n)}$ y $\{M(v)\}_v$ son computacionalmente indistinguibles

  □

Presentamos a continuación un protocolo Fiat-Shamir más simple que el anterior, donde $k = 1$ y la fase de preparación es más sencilla. En este protocolo y en el siguiente nos encontramos con que no es el centro de confianza el que emite la tarjeta con la información, sino que es el propio probador el que escoge el secreto que se encontrará en su tarjeta. Aún así, el centro de confianza supervisa que los valores tomados posean las características necesarias.

El interés de este protocolo está en que es una versión más sencilla del Protocolo 4.1 que ya no se implementa pero sirve de guía para otros protocolos y para comprender el conocimiento cero.

Protocolo 4.2 (**Fiat Shamir simplificado**). [14] El probador, A, quiere demostrar su identidad a B.

a) *Preparación.* La entidad de confianza selecciona y publica n, donde $n = pq$ con p y q números primos aleatorios que el centro de confianza no desvela.

   El probador A escoge un $s \in \mathbb{Z}_n^*$ aleatorio. Calcula $v = s^2 \bmod n$, el centro de confianza supervisa que se realizan bien las elecciones y emite la tarjeta para A, la cual contiene su clave pública, v, y su clave privada, s.

b) *Intercambio de mensajes.* Estos pasos se repiten t veces.

   1.- $A \to B$: A escoge un r de modo que $1 \leqslant r \leqslant n-1$, calcula $x = r^2 \bmod n$ y envía x a B.

   2.- $B \to A$: B escoge aleatoriamente un $e = 1$ ó $e = 0$ y se lo envía a A.

   3.- $A \to B$: A calcula $y = rs^e \bmod n$ y se lo envía a B.

   4.- Cuando B recibe y, comprueba que $y^2 = xv^e \bmod n$, acepta y pasa a la siguiente iteración. En el caso de que no se cumpla la igualdad, rechaza y para.

c) *Final del protocolo.*

   Si se completan las t iteraciones con éxito entonces B acepta y para. En el caso en el que alguna de ellas no se cumpliera, entonces se rechaza y se para.

Como en el Protocolo 4.1, también se escoge $kt = 20$ por lo que como en este caso $k = 1$, se toma $t = 20$.

El siguiente protocolo es una evolución del Protocolo 4.1 presentada por Feige, Fiat y Shamir en la que la fase de preparación varía bastante. La elección de los primos secretos p y q, y la de los secretos $s_1, ..., s_k$ se hace de manera distinta.

Protocolo 4.3 (**Feige Fiat Shamir**). [7] El probador, A, demuestra a B su identidad.

a) *Preparación.* Se fijan los valores k y t. La entidad de confianza realiza lo siguiente: escoge dos primos aleatorios congruentes con $3 \bmod 4$, p y q. Calcula $n = pq$ y publica n[1].

   El probador, A, realiza lo siguiente:

[1] El entero n que cumple estas características es conocido como un entero de Blum [14].

- Selecciona k enteros aleatorios $s_1, ..., s_k$ de modo que $s_i \in \mathbb{Z}_n^*$ con $i = 1, ..., k$. Estos s_i van a ser el secreto de A, $(s_1, ..., s_k)$ es la tarjeta de A.
- Calcula $v_i = \pm(s_i^2)^{-1}$ mód n con $i = 1, ..., k$ escogiendo + ó − aleatoriamente.
- $v_1, ..., v_k$ se convierte en la clave pública de A.

La realización de estos pasos por parte de A son supervisadas por el centro de confianza para garantizar que cumplen las características necesarias.

b) *Intercambio de mensajes.* Los siguientes pasos se realizan t veces:

1.- $A \to B$: A escoge un entero r, con $1 \leqslant r \leqslant n-1$, calcula $x = \pm r^2$ mód n y se lo envía a B, escogiendo aleatoriamente + ó −.

2.- $B \to A$: B escoge un vector de bits aleatorios $e_1, ..., e_k$ y se lo envía a A.

3.- $A \to B$: A calcula $y = r \prod_{i=1}^{k} s_i^{e_i}$ mód n, y se lo envía a B.

4.- B calcula $z = y^2 \prod_{i=1}^{k} v_i^{e_i}$ mód n y comprueba si $z = \pm x$. Si se cumple, entonces B acepta y pasa a la siguiente iteración. Si no se cumple, entonces B rechaza y para.

c) *Final del protocolo.* Si se completan las t iteraciones con éxito, entonces B acepta la identidad de A.

Observación 4.2. Algunas observaciones sobre el Protocolo 4.3:

- Se escogen $kt = 30$ por motivos de seguridad donde normalmente $k = 6$ y $t = 5$ [14] para que el protocolo sea robusto.
- Cuando B hace la comprobación en el paso 4 nos encontramos con que z puede ser igual al valor positivo o negativo de x, por lo tanto, la robustez del protocolo se vería afectada pero para ello se toman valores k y t más altos.
- A puede ocultar eficazmente la clave privada, puesto que no es factible extraer raíces cuadradas módulo n. Además A podrá demostrar su identidad demostrando el conocimiento de estos secretos s_i. Al escoger los + ó − aleatoriamente nos aseguramos que v_i abarque todos los números módulo n con símbolo de Jacobi +1 y por ello s_i existe desde el punto de vista de B.
- La elección de un entero de Blum n es para que uno de estos $+(v_i)^{-1}$ y $-(v_i)^{-1}$ tenga raíz cuadrada.

Vamos a probar de una manera menos detallada que el Protocolo 4.3 es una prueba de conocimiento cero computacional. Las ideas de la demostración son las mismas y por lo tanto muchos de los detalles se obviarán.

Proposición 4.2. El protocolo 4.3 es una prueba de conocimiento cero computacional.

Demostración. Probaremos que es un protocolo completo, robusto y de conocimiento cero computacional:

- *Completitud.* Supongamos que A conoce los $s_1, ..., s_k$ y en el paso 3 ha enviado $y = r\prod_{i=1}^{k} s_i^{e_i}$ mód n a B. En el paso 4, B siempre obtendrá que $z = \pm x$ por como están construidos los $v_1, ..., v_k$. Por ello, el protocolo es robusto.

- *Robustez.* Supongamos que un impostor A^* cualquiera no conoce los $s_1, ..., s_k$ y quiere hacerse pasar por A, para ello, tendría que escoger los $e_1, ..., e_k$ que fuera a elegir B en el paso 2 y además, tomar y aleatorio y escoger $x = y^2\prod_{i=1}^{k}$ módn. La probabilidad de que acierte la cadena de bits e_i es de 2^{-k}, y en total en el protocolo 2^{-kt}. Como hemos visto en la Observación 4.2, la probabilidad de que B acepte la identidad de un impostor es 2^{-30}, un valor ínfimo y por lo tanto, el protocolo es robusto.

- *Conocimiento cero computacional.* La demostración de que es un protocolo de conocimiento cero computacional es análoga a la de la Proposicion 4.2.

□

A continuación compararemos los protocolos vistos en esta sección:

- El centro de confianza es común a los tres protocolos pero en cada uno realiza funciones distintas: en el Protocolo 4.1 es el centro de confianza el que otorga los secretos al probador, mientras que en los otros dos, es el propio probador el que escoge los valores. En todos los protocolos hace labor de supervisión y desaparece después de que son adjudicadas las claves públicas y privadas.

- El Protocolo 4.1 es el único que incluye el uso de funciones DES y es también el más arcaico. Este tipo de funciones hoy en día han sido sustituidas por otras por razones de seguridad.

4.3. Ejemplos

Los siguientes ejemplos se realizan con valores más pequeños que los que se tomarían en una implementación real. Su único fin es clarificar los protocolos de identificación expuestos en esta sección. Los cálculos y la búsqueda de números primos se ha hecho a través de la herramienta Mathematica. Se incluyen ejemplos de los Protocolos 4.1, 4.2 y 4.3

Ejemplo 4.1. Utilizando las mismas notaciones que en el Protocolo 4.1:

a) *Preparación.* Se fijan los enteros $k = 2$ y $t = 1$. El centro de confianza escoge dos números primos $p = 37$ y $q = 59$, y publica $n = pq = 2183$.

Publica una función DES f (hemos utilizado una que proporciona Mathematica). Cuando A se identifica al centro de confianza, éste genera la cadena $I = 29$. Después, el centro de confianza realiza lo siguiente para poder emitir la tarjeta:

- Calcula un número indeterminado de $v_j = f(I, j)$ y escoge los que son un residuo cuadrático módulo n. Por simplificar notación: $j = 1, 2$, luego $v_1 = 100$ y $v_2 = 108$ (a través de Mathematica se ha comprobado que son residuos cuadráticos).
- Calcula $s_1 = v_1^{-1/2}$ mód n y $s_2 = v_2^{-1/2}$ mód n y obtenemos que $s_1 = 655$ mód n y $s_2 = 548$ mód n.
- El centro de confianza emite la tarjeta electrónica $(I; s_1, s_2)$ para A.

▪ *Intercambio de mensajes*

1.- $A \to B$: A envía $I = 29$ y $j = 1, 2$ a B.

2.- $B \to A$: B calcula los v_1, v_2.

3.- $A \to B$: A escoge $r = 327$, calcula $x = r^2$ mód $n = 2145$ mód 2183 y envía $x = 2145$ mód 2183 a B.

4.- $B \to A$: B escoge como vector aleatorio $(0, 1)$ y se lo envía a A.

5.- $A \to B$: A calcula $y = rs_2 = 327 \cdot 548 = 190$ mód 2183 y se lo envía a B.

6.- B calcula $y^2 = 1172$ mód 2183 e $y^2 v_2 = 1172 \cdot 108 = 2145$ mód 2183 y comprueba que $y^2 v_2 = x$.

c) *Finalización del protocolo* Como los pasos solo se realizan una vez porque $t = 1$ entonces B acepta la identidad de A.

Ejemplo 4.2. Utilizando las mismas notaciones que en el Protocolo 4.2

▪ *Preparación.* El centro de confianza escoge los números primos $p = 379$ y $q = 787$. Calcula $n = pq$ y publica el resultado $n = 298273$.

El probador, A, toma un $s \in \mathbb{Z}^*_{298273}$. Escoge $s = 127085$ y calcula $v = s^2$ mód n. Calcula $s^2 = 16150597225 = 9094$ mód 298273 y publica $v = 9094$ mód 298273 como su clave pública.

▪ *Intercambio de mensajes.* Los siguientes mensajes se realizarían t veces, como es una ejemplo sencillo, tomaremos $t = 2$.

$t = 1$:

- $A \to B$: A toma un r aleatorio con $1 \leq r < 298273$ y escoge $r = 25868$. Calcula $x = r^2$ mód $n \Rightarrow r^2 = 669153424 = 127085$ mód 298273 y envía $x = 127085$ mód 298273 a B.
- $B \to A$: B escoge aleatoriamente $e = 0$ ó $e = 1$. En este caso escoge $e = 1$ y se lo envía a A.
- $A \to B$: A calcula $y = rs^e$ mód $n = rs$ mód $n \Rightarrow y = rs = 25868 \cdot 127085 = 3287434780 = 168047$ mód 298273. A envía $y = 168047$ mód 298273 a B.

- B recibe y, calcula y^2 mód n por un lado y xv^e mód n, por otro. $y^2 = 28239794209 = 201388$ mód 298273 y $xv = 127085 \cdot 9094 = 1155710990 = 201388$ mód 298273. Como vemos $y^2 = xv$ por lo tanto B acepta y pasamos a la siguiente iteración.

$t = 2$:

- $A \to B$: A toma un r aleatorio con $1 \leq r < 298273$ y escoge $r = 29$. Calcula $x = r^2$ mód $n \Rightarrow r^2 = 841$ mód 298273 y envía $x = 841$ mód 298273 a B.
- $B \to A$: B escoge aleatoriamente $e = 0$ ó $e = 1$. En este caso escoge $e = 0$ y se lo envía a A.
- $A \to B$: A calcula $y = rs^e$ mód $n \Rightarrow y = r$ porque $e = 0$. Luego, como $r = 29$, entonces $y = 29$ mód 298273. A envía $y = 29$ mód 298273 a B.
- B recibe y, calcula y^2 mód n por un lado y xv^e mód $n = x$ mód n, por otro. $y^2 = 841$ mód 298273 y $x = 841$ mód 298273. Como vemos $y^2 = xv$ por lo tanto B acepta.

- *Finalización del protocolo*

  Como se han completado las dos iteraciones con éxito, entonces B aceptará la identidad de A.

Ejemplo 4.3. Utilizando las mismas notaciones que en el Protocolo 4.3. En este caso los enteros seguros que se toman son $k = 3$ y $t = 1$ para simplificar el ejemplo, pero como vimos en el Protocolo 4.3 lo mejor es $kt = 20$.

- *Preparación.* El centro de confianza escoge primos aleatorios congruentes a 3 mód 4, $p = 683$, $q = 811$. Calcula $n = pq$ y publica el entero Blum $n = 553913$. El probador, A, toma tres enteros aleatoriamente $s_i \in \mathbb{Z}^*_{553913}$ con $i = 1, 2, 3$. A escoge $s_1 = 157$, $s_2 = 43215$, $s_3 = 4646$. Después, calcula los v_i y elige aleatoriamente el signo: $v_1 = 441845$, $v_2 = 338402$, $v_3 = 124423$. Los v_i para $i = 1, 2, 3$ serán la clave pública de A.

- *Intercambio de mensajes.*

  - $A \to B$: A toma un r aleatorio con $1 \leq r < n$. Escoge $r = 4527$, un símbolo $+$ ó $-$ y calcula $x = \pm r^2$ mód $n \Rightarrow x = 20493729 = +552861$ mód 553913. Envía a B el valor $x = +552861$ mód 553913
  - $B \to A$: B toma un vector de bits aleatorios (e_1, e_2, e_3). Escoge $(0, 1, 0)$ y se lo envía a A.
  - $A \to B$: A calcula $y = r(s_1^{e_1} s_2^{e_2} s_3^{e_3}) \Rightarrow y = rs_2 \Rightarrow y = 4527 \cdot 43215 = 195634305 = 103016$ mód 553913 y envía a B $y = 103016$ mód 553913.
  - El verificador ,B, calcula $z = y^2 \cdot (v_1^{e_1} v_2^{e_2} v_3^{e_3})$, como el vector binario es $(0, 1, 0)$, entonces $z = y^2 \cdot v_2$ mód n. $y^2 = 10612296256 = 431002$ mód 553913 y $z = y^2 \cdot v_2 = 431002 \cdot 338402 = 145851938804 = 552861$ mód 553913. Y vemos que $z = +x$ porque $z = 552861$ mód 553913 y $x = 552861$ mód 553913.

- *Finalización del protocolo.* Como se cumple que $z = x$, entonces B acepta la identidad de A.

Capítulo 5

Pruebas de conocimiento cero no interactivas y firmas digitales

En este capítulo presentamos las pruebas no interactivas de conocimiento cero y las firmas digitales. Hasta ahora, habíamos visto las pruebas de conocimiento cero interactivas en las que el verificador proponía una especie de desafío al probador y éste respondía. En este caso, el probador envía un solo mensaje al verificador que le permite demostrar su conocimiento sobre algo.

Las firmas digitales pueden pensarse como pruebas de conocimiento cero no interactivas en las que simplemente se entrega un mensaje que demuestra la autoría del probador o pueden ser una variación de una prueba interactiva de conocimiento cero.

En la Sección 5.2.3 observaremos tres tipos de firmas digitales: una de ellas está basada en la prueba de conocimiento cero no interactiva de la Sección 5.1 y las otras dos, en protocolos interactivos de identificación.

5.1. Pruebas de conocimiento cero no interactivas

En esta sección abordaremos las principales nociones de las pruebas no interactivas de conocimiento cero y un protocolo de este tipo. Volvemos al problema de la pertenencia de un elemento a un lenguaje visto en los Capítulos 2 y 3, solo que ahora el planteamiento y la solución que se da es distinta.

En las pruebas de conocimiento no interactivas no hay intercambio de mensajes entre el verificador y el probador. El probador envía solo una cadena de información al verificador que le permite comprobar que dice la verdad pero sigue existiendo un componente probabilístico que explicaremos más adelante. Si pensamos en el ejemplo de la Sección 1.1, Paquita enviaría a los periodistas un mensaje que les permitiría saber que ella conoce la palabra secreta que abre el pasadizo entre los dos pueblos, no habría intercambio de mensajes de ningún tipo.

Blum, Feldman y Micali en [3] introducen las pruebas de conocimiento cero no interactivas y el problema que tratan es el mismo que en las pruebas interactivas: el probador debe demostrar al verificador la pertenencia de un elemento a un lenguaje. La diferencia con las pruebas interactivas es que el probador envía al verificador solo una cadena con información, la cual le permite verificar que el probador sabe por qué el elemento pertenece

a ese lenguaje.

Todas las definiciones y notaciones de máquinas de Turing de la Sección 2.1 serán utilizadas para formalizar la definición de prueba de conocimiento cero no interactiva.

Los conceptos de completitud y robustez (Sección 2.2) sufren modificaciones, pero la idea principal se mantiene: si el elemento está en el lenguaje, el probador convence al verificador con una probabilidad cercana a 1 (completitud); y si el elemento no está en el lenguaje, el probador convence al verificador con una probabilidad muy cercana a 0 (robustez). En cuanto al conocimiento cero, el planteamiento es el mismo: la vista del verificador durante el protocolo puede ser simulada por una máquina de Turing probabilística ajena a la prueba.

La principal diferencia con las pruebas interactivas es que el probador tiene acceso a la cinta aleatoria del verificador. La garantía de que los protocolos sean de conocimiento cero está en que el contenido de esta cinta sea aleatorio, es decir, que no se manipule.

Por el Capítulo 2, sabemos que una prueba de conocimiento cero es un sistema de pruebas de conocimiento cero, por lo tanto, vamos a definir un sistema de pruebas no interactivas de conocimiento cero. Para ello, necesitamos redefinir la completitud y la robustez.

Para formalizar la definición de prueba no interactiva, recurriremos como en el Capítulo 2 a las máquinas de Turing y redefiniremos al probador y al verificador como máquinas de Turing:

Definición 5.1. El probador, A, en una prueba de conocimiento cero no interactiva es una máquina de Turing probabilística con cinco cintas:

- Una cinta de entrada de solo lectura.
- Una cinta de trabajo de lectura y escritura.
- Una cinta aleatoria de solo lectura.
- Una cinta aleatoria adicional de solo lectura.
- Una cinta comunicación de solo escritura.

Definición 5.2. El verificador, B en una prueba de conocimiento cero no interactiva es una máquina de Turing probabilística con cinco cintas:

- Una cinta de entrada de solo lectura.
- Una cinta de trabajo de lectura y escritura.
- Una cinta aleatoria de lectura.
- Una cinta de comunicación de solo lectura.
- Una cinta de salida de solo escritura.

Definición 5.3. Las dos máquinas de Turing de las Definiciones 5.1 y 5.2 forman un par de máquinas no interactivas si cumplen lo siguiente:

- La cinta de entrada es común a ambas.
- La cinta de comunicación de ambas es la misma. El probador solo escribe y el verificador solo lee.
- La cinta aleatoria del verificador es también visible para el probador.

Al par de máquinas de Turing que cumple estas características lo denotamos por (A, B) que es la misma notación que para los sistemas de pruebas interactivas, por lo que siempre señalaremos si da lugar a dudas, si se trata de una prueba interactiva o no interactiva.

Observación 5.1. Si comparamos con las pruebas interactivas que habíamos visto hasta el Capítulo 3, nos encontramos con que la máquina de Turing que representa al verificador es distinta a la del probador: en el verificador la cinta de comunicación es de solo lectura y en el probador, de solo escritura; y la cinta aleatoria del verificador, es también visible para el probador.

Recordemos que si el verificador acepta que un elemento está en el lenguaje devuelve un 1, si no un 0.

Definición 5.4. Sea (A, B) un par de máquinas de Turing no interactivas y L un lenguaje, denotamos por $[A(x), B(x)]$ a la distribución de salida de B cuando tiene como valor de entrada el elemento $x \in L$. Esta distribución está inducida por las cintas aleatorias de las dos máquinas.

Definición 5.5. Un par de máquinas de Turing como en la Definición 5.3 es un sistema de pruebas no interactivas si el probador no tiene límite computacional, el verificador tiene tiempo esperado polinomial en función del valor de entrada y se verifican las siguientes propiedades:

- *Completitud*. Sea un $x \in L$ suficientemente grande se cumple que para cada $c > 0$:

$$P([A(x), B(x)] = 1) > 1 - |x|^{-c} \tag{5.1}$$

- *Robustez*. Sea un $x \notin L$ suficientemente grande se cumple que para cada $c > 0$:

$$P([A(x), B(x)] = 0) > 1 - |x|^{-c} \tag{5.2}$$

Una vez vista la definición de sistema de pruebas no interactivas vamos a definir un sistema de pruebas no interactivas de conocimiento cero, para ello necesitamos las definiciones y notaciones sobre conocimiento cero de la Sección 2.3 que no sufren modificaciones.

Definición 5.6. Sea (A, B) un sistema de pruebas no interactivas, sea $\{\langle A, B\rangle(x)\}_{x\in L}$ la vista del verificador B durante la prueba que tiene como valor de entrada un $x \in L$. Decimos que (A, B) es un sistema de pruebas no interactivas de conocimiento cero perfecto (resp. estadístico, resp. computacional) en L si la familia de variables aleatorias $\{\langle A, B\rangle(x)\}_{x\in L}$ es perfectamente (resp. estadísticamente, resp. computacionalmente) aproximable en L

A continuación mostramos el sistema de pruebas no interactivas que se propone en [3] que es una prueba de conocimiento cero no interactiva sobre el lenguaje $G3C_k = \{G(V,E) \in G3C : |V| \leq k\}$. Para el desarrollo de este protocolo se necesitarán las nociones de residuos cuadráticos dadas en la Sección 3.3 y utilizaremos el conjunto $\mathbb{Z}_p^{+1} = \{q \in \mathbb{Z}_p : \left(\frac{q}{p}\right) = +1\}$.

Protocolo 5.1. Puede consultarse en [3]. El probador, A, quiere demostrar al verificador, B, que un grafo $G(V,E)$ está en $G3C_k$ sin aportarle una coloración correcta. Ambos conocen $G(V,E)$. Solo A conoce una coloración correcta de $G(V,E)$.

a) *Preparación.* El probador realiza las siguientes acciones:

1.- Escoge aleatoriamente n_1, n_2, n_3 de modo que cada n_i sea producto de dos primos.

2.- Escoge aleatoriamente q_1, q_2, q_3 de modo que $\left(\frac{q_i}{n_i}\right) = 1$ y q_i no sea un residuo cuadrático módulo n_i para $i = 1, 2, 3$.

3.- Se asignan los colores correspondientes $\{1, 2, 3\}$ al grafo G.

4.- A cada vértice del grafo con coloración i para $i = 1, 2, 3$ se le asigna una terna $(v_1, v_2, v_3) \in \mathbb{Z}_{n_1}^{+1} \times \mathbb{Z}_{n_2}^{+1} \times \mathbb{Z}_{n_3}^{+1}$ distinta, donde se verifica que $Q_{n_i}(v_i) = 0$ y $Q_{n_j}(v_i) = 1$ para $j \neq i$. Al grafo con las nuevas asignaciones lo denominamos G'.

5.- Se escogen un número indefinido de ternas aleatorias $(z_1, z_2, z_3) \in \mathbb{Z}_{n_1}^{+1} \times \mathbb{Z}_{n_2}^{+1} \times \mathbb{Z}_{n_3}^{+1}$ y se asignan $8k$ a cada una de las aristas del grafo G.

6.- Para cada arista (a, b) del grafo G' (donde $a = (a_1, a_2, a_3)$ y $b = (b_1, b_2, b_3)$) y para cada una de las $8k$ ternas asignadas a cada arista, (z_1, z_2, z_3), podremos computar solo una de las siguientes firmas:

I.- $(\sqrt{z_1}, \sqrt{z_2}, \sqrt{z_3})$. En el caso en el que cada z_i sea un residuo cuadrático módulo n_i para $i = 1, 2, 3$.

II.- $(\sqrt{q_1 z_1}, \sqrt{z_2}, \sqrt{z_3})$. En el caso en el que z_1 no sea un residuo cuadrático módulo n_1 y los otros z_i si sean un residuo cuadrático módulo n_i para $i = 2, 3$.

III.- $(\sqrt{z_1}, \sqrt{q_2 z_2}, \sqrt{z_3})$. Lo análogo al tipo II para z_2.

IV.- $(\sqrt{z_1}, \sqrt{z_2}, \sqrt{q_3 z_3})$.Lo análogo al tipo II para z_3.

V.- $(\sqrt{a_1 z_1}, \sqrt{a_2 z_2}, \sqrt{a_3 z_3})$. En el caso en el que dos de los z_i no sean residuos cuadráticos módulo n_i.

VI.- $(\sqrt{b_1 z_1}, \sqrt{b_2 z_2}, \sqrt{b_3 z_3})$.En el caso en el que dos de los z_i no sean residuos cuadráticos módulo n_i

VII.- $(\sqrt{a_1 b_1 z_1}, \sqrt{a_2 b_2 z_2}, \sqrt{a_3 b_3 z_3})$.En el caso en el que dos de los z_i no sean residuos cuadráticos módulo n_i.

VIII.- $(\sqrt{q_1 z_1}, \sqrt{q_2 z_2}, \sqrt{q_3 z_3})$. En el caso en el que ningún z_i sea residuo cuadrático módulo n_i para $i = 1, 2, 3$.

b) *Mensaje enviado.* El probador envía al verificador los $n_1, n_2, n_3, q_1, q_2, q_3, G'$ y las firmas de cada una de las ternas que hemos obtenido del paso 6.

c) *Final de la prueba.* El verificador comprueba que los n_1, n_2, n_3 no son potencia de ningún entero y que se cumple que $(v_1, v_2, v_3) \in \mathbb{Z}_{n_1}^{+1} \times \mathbb{Z}_{n_2}^{+1} \times \mathbb{Z}_{n_3}^{+1}$. También comprueba que las firmas del paso 6 están bien hechas.

Si se cumplen estas condiciones entonces el verificador acepta que el grafo G es 3-coloreable

Observación 5.2. Ahora veamos unas observaciones sobre el Protocolo 5.1:

- En el paso 4, se asignan ternas a cada vértice de modo que para un vértice de color i el elemento v_i de la terna sea un residuo cuadrático módulo n_i.
- En el paso 5, los z_i se escogen utilizando la cinta aleatoria que es común a probador y verificador.
- La elección de $8k$ ternas para cada arista es porque hay 8 tipos de firma y así se logra que el protocolo sea más seguro, y por consiguiente, robusto [3].
- En cuanto a la verificación de la firma, recibe el grafo G' pero no puede acceder al color de los vértices porque no puede comprobar si un elemento es residuo cuadrático, ya que no conoce la descomposición de los n_i. Para comprobar las $8k$ firmas de cada arista, comprueba que el cuadrado de cada elemento de las ternas puede ser divisible por algún q_i(firmas II, III, IV, VIII) o alguno de los v_i(firmas V, VI, VII). Alguno podría tener la firma I, pero es por ello que se envían tantas firmas, para que la probabilidad de que se tome siempre la firma I sea ínfima.

5.2. Firmas digitales

En esta sección presentamos tres protocolos de firmas digitales. Cada uno está enfocado de una manera distinta pero todos están basados en pruebas de conocimiento cero interactivas y no interactivas.

5.2.1. Firma digital de Fiat y Shamir

Fiat y Shamir [8] proponen esta firma digital que está basada en el Protocolo 4.1 de identificación del capítulo anterior.

Como decíamos en la Sección 4.1, las tarjetas electrónicas que emite el centro de confianza además de permitir encriptar y desencriptar, también permiten firmar y verificar mensajes. Para este protocolo de firma partimos desde la fase previa del Protocolo 4.1, donde el probador A recibe del centro de confianza la tarjeta electrónica.

Protocolo 5.2 (**Firma Fiat-Shamir** [8])**.** Supongamos que A quiere firmar un mensaje m, mandárselo a B y que B verifique que la firma es de A.

a) *Preparación.* Se fijan los enteros k y t. El centro de confianza realiza los siguientes pasos:

- Escoge dos números primos secretos p y q, y publica su producto $n = pq$.
- Publica una función DES, f que asigna cadenas arbitrarias a valores en $\mathbb{Z}_n$.

El probador se identifica al centro de confianza mediante métodos no criptográficos. Después, el centro crea una cadena I que contiene información sobre el usuario(nombre, dirección, correo) y sobre la tarjeta(fecha de espiración, limitaciones de validez). Para crear esta tarjeta realizan los siguientes pasos:

- Calcula un número indeterminado de $v_j = f(I, j)$ para $j = 0, 1, ...$
- De todos los v_j escoge k cualesquiera de modo que v_j es un residuo cuadrático módulo n. Para simplificar la notación tomaremos $j = 1, ..., k$.
- Calcula s_j que es la raíz cuadrada más pequeña de v_j^{-1} mód n para $j = 1, ..., k$.
- El centro emite como tarjeta para A la cadena $(I; s_1, ..., s_k)$. Los s_j son el secreto de A.

El firmante, A, realiza los siguientes pasos:

1.- Escoge enteros aleatoriamente r_i de modo que $1 \leq r_i < n$. Calcula $x_i = r_i^2$ mód n para $i = 1, ..., t$.

2.- Calcula $f(m, x_1, ..., x_t)$ con f la función DES y toma los kt primeros bits como valores e_{ij} con $1 \leq i \leq t, 1 \leq j \leq k$.

3.- Calcula y_i para $i = 1, ..., t$:

$$y_i = r_i \prod_{e_{ij}=1} s_j \text{ mód } n \tag{5.1}$$

b) *Envío de firma. A* envía I, m, la matriz e_{ij}, y todos los y_i a B.

c) *Comprobación de firma. B* calcula $v_j = f(I, j)$ para $j = 1, ..., k$,
$z_i = y_i^2 \prod_{e_{ij}=1} v_j$ mód n para $i = 1, .., t$ y comprueba que los primeros kt bits de $f(m, z_1, ..., z_t)$ son los e_{ij}.

Para obtener este protocolo de firma se ha cambiado el papel que desempeña el verificador B en el Protocolo 4.1 por la función DES f.

En esta firma,para conseguir un nivel de seguridad alto, se tiene que aumentar como mínimo a $kt = 72$ [8] y no $kt = 20$ (Protocolo 4.1 de identificación).

5.2.2. Firma digital de Bellarre y Goldwasser

Bellare y Goldwasser [2] presentan un protocolo de firma digital basado en las pruebas no interactivas de conocimiento cero y las funciones pseudoaleatorias.

Definición 5.7. Una función pseudoaleatoria es una función cuya imagen es computacionalmente fácil de calcular y para la que no existe un algoritmo de tiempo polinomial probabilístico que la diferencie de una función aleatoria [15].

Las pruebas de conocimiento cero no interactivas que definen están basadas en las vistas en la Sección 5.1 de [3] pero incluyendo algunos cambios: la prueba se divide en dos fases: la fase preparatoria y la de demostración. En la primera fase se establece alguna información común para probador y verificador (en relación a las pruebas de la Sección 4.1 esa información común sería la cadena de bits aleatorios σ), e información privada para cada uno . En la segunda fase, el probador demuestra al verificador que conoce una información sin revelarla, utilizando la información de la primera etapa. El proceso de la primera etapa es independiente de lo que vaya a probarse en la segunda etapa.

Nota 5.1. Una colección de funciones pseudoaleatorias la denotamos así $F_k = \{f_s : |s| = k\}$.

Para la firma digital de Bellarre y Goldwasser basada en su propuesta de prueba no interactiva de conocimiento cero [2], la fase previa la realizaría un centro de confianza y su funcionamiento sería el siguiente: el firmante (probador) recibe del centro de confianza una tarjeta electrónica que le permite firmar los mensajes.

Antes de pasar al protocolo haremos algunas aclaraciones: para este protocolo es necesaria también una prueba no interactiva de conocimiento cero verificable públicamente para algún lenguaje NP; y, como ya hemos aclarado antes, la fase previa del protocolo para firma digital es sustituida por un centro de confianza al que se identifican los usuarios y éstos reciben las claves para firmar y verificar.

Nota 5.2. Sea (A, B) una prueba de conocimiento cero no interactiva denotamos $(A, B)_\gamma$ a la prueba de conocimiento cero no interactiva que tiene γ como cinta aleatoria común.

Nota 5.3. Denotamos E a un algoritmo cualquiera de encriptación de clave pública probabilística.

Protocolo 5.3 (**Bellarre Goldwasser** [2])**.** El probador A, quiere firmar un mensaje m y B debe verificarlo.

a) *Preparación.* Todos los participantes en la firma obtienen del centro de confianza la clave pública $(1^k, E, \alpha, \gamma)$, su clave privada (r, s) y la función pseudoaleatoria f_s:

   1.- k es un entero seguro.

   2.- $s \in_R \{0, 1\}$ es el índice de la función pseudoaleatoria.

   3.- $\alpha = E(r, s)$ es una encriptación de s.

   4.- γ es la información común de la prueba (A, B).

   El firmante calcula $R = f_s(m)$

b) *Envío de firma.* El firmante envía a B la cadena $(m, R, (A, B)_\gamma)$.

c) *Comprobación de firma.* El verificador comprueba con la clave pública del firmante que se verifican tanto la prueba como la firma. Comprueba que $f_s(m) = R$ para la firma.

5.2.3. Firma digital de Schnorr

Schnorr propone el protocolo de firma digital que presentamos en esta subsección [16]. En este protocolo el centro de confianza tiene menos presencia que en los anteriores porque es el propio firmante el que escoge su clave privada y su clave pública.

Protocolo 5.4 (**Schnorr** [16])**.** El probador, A, quiere firmar un mensaje m para el verificador, B.

a) *Preparación.* Se fija el entero t. El centro de confianza lleva a cabo los siguientes pasos:

1.- Escoge p y q de modo que q sea divisor de $p-1$ y $q \geq 2^{2t}$, $p \geq 2^{7t}$.

2.- Elige un $\alpha \in \mathbb{Z}_p$ de orden q.

3.- Selecciona una función hash $h : \mathbb{Z}_q \times \mathbb{Z} \to \{0, ..., 2^t - 1\}$.

El centro de confianza publica la cadena (p, q, α, h).

El firmante realiza las siguientes acciones:

4.- Escoge un $s \in \mathbb{Z}_q$ como su clave privada y obtiene su clave pública calculando $v = \alpha^{-s}$ mód p. Publica v como su clave pública.

5.- Elige $r \in_R \mathbb{Z}_q$ y calcula $x = r^2$ mód p.

6.- Calcula $e = h(m, x)$.

7.- Calcula $y = r + se$ mód q.

b) *Envío de firma.* A publica la siguiente cadena como firma del menasaje m: (e, y).

c) *Comprobación de firma.* B calcula $x' = \alpha^y v^e$ mód p y comprueba que $x' = x$. Si es así, acepta la firma de A como válida.

Observación 5.3. Respecto al Protocolo 5.4 podemos hacer las siguentes observaciones:

- En la preparación se fija un entero t que por razones de seguridad debe ser $t = 72$ [16].
- Como vemos en el paso 4, obtener la clave pública a partir de la privada es computacionalmente fácil, mientras que la inversa es el problema del logaritmo discreto, por lo tanto difícil de resolver.

5.3. Ejemplos

Los ejemplos que se presentan en esta sección utilizan valores más pequeños que en implementaciones reales. Su única labor es clarificar un poco más el protocolo no interactivo de la Sección 5.1 y mostrar los protocolos de firma de la sección anterior . Todos los cálculos, búsqueda de número primos, funciones DES y hash se obtienen utilizando la herramienta Mathematica.

Ejemplo 5.1. Utilizando las mismas notaciones que en el Protocolo 5.1 vamos a mostrar parcialmente cómo se realizaría dicho protocolo. La idea es que el probador debe demostrar al verificador que el grafo $G \in G3C_4$ sin mostrarle una coloración. Tanto el probador como el verificador conocen el grafo. El probador es el único que conoce la coloración del grafo. Se procede de la siguiente manera:

1.- Escoge $n_1 = 79 \cdot 29, n_2 = 47 \cdot 23, n_3 = 97 \cdot 11$, de modo que $n_1 = 2291, n_2 = 1081, n_3 = 1067$.

2.- Elige $q_1 = 12, q_2 = 5, q_3 = 17$.

3.- Asignamos los colores $1, 2, 3$ correspondientes a cada vértice del grafo G.

4.- Vamos a asignar ternas a cada vértice, en este caso asignaremos solo a dos vértices para hacer el ejemplo, escogemos un vértice con el color 1 y otro con el color 2. Al vértice de color 1 le asignamos la terna $(22, 30, 21) \in \mathbb{Z}_{2291}^{+1} \times \mathbb{Z}_{1081}^{+1} \times \mathbb{Z}_{1067}^{+1}$ donde 22 es un residuo cuadrático módulo n_1 y el resto no es un residuo cuadrático módulo n_j con $j = 2, 3$. De forma análoga, asignamos otra terna al vértice de color 2 adyacente al anterior. La terna que asociamos es $(36, 43, 49)$ donde 43 es un residuo cuadrático módulo 1081. Al grafo con todas las ternas asignadas lo denominamos G'.

5.- Tendríamos que tomar $8 \cdot 4 = 32$ ternas para cada arista del grafo, en este caso tomaremos dos ternas para la arista que une los vértices que hemos renombrado en el paso anterior. Escogemos $(13, 20, 103)$ y $(45, 24, 64)$ ambas pertenecientes a $\mathbb{Z}_{2291}^{+1} \times \mathbb{Z}_{1081}^{+1} \times \mathbb{Z}_{1067}^{+1}$.

Procedemos ahora a firmar estas dos ternas:

- La terna $(13, 20, 103)$ la firmamos con VI utilizando $(36, 43, 49)$, porque dos de los componentes de la terna no son residuos cuadráticos y por lo tanto la terna ya firmada queda como $(288, 72, 107)$.
- La terna $(45, 24, 64)$ la firmamos con I, porque todos los componentes de la terna son residuos cuadráticos módulo su n_i correspondiente. La terna ya firmada es $(613, 114, 8)$.

El probador envía al verificador la cadena
$(n_1, n_2, n_3, q_1, q_2, q_3, G') = (2291, 1081, 1067, 12, 5, 17, G')$ y las ternas firmadas, en este caso, $(288, 72, 107)$ y $(613, 114, 8)$.

El verificador comprueba que los $n_1 = 2291, n_2 = 1081, n_3 = 1067$ no son potencias de enteros, que las ternas de los vértices cumplen que $(22, 30, 21)$ y $(36, 43, 49)$ están en $\mathbb{Z}_{2291}^{+1} \times \mathbb{Z}_{1081}^{+1} \times \mathbb{Z}_{1067}^{+1}$ y que las ternas firmadas están firmadas con alguna de

las firmas dadas. Una vez están comprobadas, entonces el verificador admite que $G \in G3C_4$.

Ejemplo 5.2. Utilizando las mismas notaciones que en el Protocolo 5.2. El probador, A quiere firmar el mensaje $m = 385$ y que B lo verifique. La fase de preparación de este protocolo es igual que en el Protocolo 4.1.

a) *Preparación.* Se fijan los enteros $k = 2$ y $t = 1$. El centro de confianza elige dos números primos $p = 37$ y $q = 59$ y publica $n = pq = 2183$. Publica también una función DES f.

El probador se identifica al centro de confianza y éste genera una cadena I. Como en el Ejemplo 4.1 se obtienen los v_i y los s_i para $i = 1, 2$, y se proporciona la tarjeta $(I; s_1, s_2)$ a A.

b) *Intercambio de mensajes.* El firmante realiza estos pasos:

1.- Escoge un entero aleatorio $r = 15$, calcula $x = r^2 = 225$ mód 2183 y envía x a A.

2.- Calcula $f(m, x) = 10100111$ y toma los $kt = 2$ primeros bits, denotándolos $e_1 = 1, e_2 = 0$.

3.- Calcula $y = rs_1^{e_1}$ donde $y = 15 \cdot 655 = 975$ mód 2183.

Después de realizar estos pasos, el probador envía al verificador la cadena $(I = 58, m = 385, (1, 0), y = 975)$

c) *Finalización del protocolo.* B calcula los v_i con $i = 1, 2$ y $z = y^2 v_1$, como $y^2 = 1020$ mód 2183, entonces $z = 1020 \cdot 100 = 1582$ mód 2183. Si calcula $f(m, z) = 10111101$ y toma los $kt = 2$ primeros dígitos, $e_1' = 1$ y $e_2' = 0$. Comprueba que $e_1 = e_{1'}$ y que $e_2 = e_2'$. Por lo tanto, B acepta la firma de A como válida.

Ejemplo 5.3. Ejemplo del Protocolo 5.4. en el que A quiere firmar el mensaje $m = 124$ y que B verifique dicha firma.

a) *Preparación.* El centro de confianza realiza los siguientes pasos:

1.- Escoge $p = 48731$ y $q = 443$ de modo que q sea divisor de $p - 1$.

2.- Elige un $\alpha = 11444$ mód 48731 de orden q.

3.- Selecciona una función hash $h : \mathbb{Z}_q \times \mathbb{Z} \to \{0, ..., 2^t - 1\}$. Utilizamos una de las funciones Hash que proporciona Mathematica.

El centro de confianza publica la cadena $(48731, 443, 11444, h)$.

El firmante realiza las siguientes acciones:

4.- Escoge $s = 89$ como su clave privada y obtiene su clave pública calculando $v = \alpha^{-s}$ mód $p = 11444^{-89} = 2974$ mód 48731. Publica $v = 2974$ mód 48731 como su clave pública.

5.- Elige $r = 312$ y calcula $x = \alpha^{-r}$ mód $p = 45300$ mód 48731.

6.- Calcula $e = h(m, x) = 14095$.

7.- Calcula $y = r + se$ mód $q = 191$ mód 443.

b) *Envío de firma.* A publica la siguiente cadena como firma del mensaje m: (e, y).

c) *Comprobación de firma.* B calcula $x' = \alpha^y v^e = 11444^{191} \cdot 2974^e = 45300$ mód 48731 y comprueba que $x' = x$. Como es así, acepta la firma de A como válida.

Capítulo 6

Una aplicación actual

En este último capítulo vamos a presentar una de las aplicaciones actuales que se le dan a las pruebas de conocimiento cero. La principal aplicación es en *blockchain*, en particular dentro de las finanzas.

Daremos unas nociones previas sobre la teoría de blockchain o cadena de bloques y la relación que tienen con ella las pruebas de conocimiento cero.

6.1. *Blockchain*

Blockchain [4] es una red única y consensuada en la que la información se distribuye en distintos bloques. Estos bloques almacenan: una cantidad de registros o transacciones, información propia del bloque y un código único que identifica a cada bloque y lo vincula con su bloque anterior y posterior. Es por esto último, que cada bloque tiene un lugar determinado dentro de la red.

Dentro de cada bloque se encuentra una copia exacta de toda la cadena, es decir, de la posición de los bloques, y a medida que se crean nuevos registros se van añadiendo a la cadena como nuevos bloques, después de ser validados por el resto de bloques de la red.

La principal ventaja de estas redes de bloques es la seguridad frente a otros métodos de almacenamiento de datos. Ningún bloque puede eliminarse porque hay una copia de la cadena en cada uno de los bloques, por lo que si un atacante quisiera eliminar un bloque debería acceder a cada uno de los bloques y borrar de la copia ese bloque.

Su relación con las pruebas de conocimiento cero viene de la seguridad que utiliza cada bloque. Cada uno de estos bloques recurre a certificados y firmas digitales para verificar la información y validar los datos almacenados en la red, lo cual asegura la autenticidad de toda la información.

Uno de los tipos de pruebas de conocimiento cero que pueden aplicarse a la seguridad de estas redes son las pruebas de rango de conocimiento cero, presentadas en [13] y que están orientadas sobre todo a finanzas.

6.2. Pruebas de rango de conocimiento cero

El protocolo que presentamos en este apartado es una prueba de conocimiento cero no interactiva (Sección 5.1) que permite al probador demostrar que un valor se encuentra

dentro de un rango sin revelar al verificador ese valor. Este protocolo cuenta también con un centro de confianza que participa en la preparación del protocolo.

Saber si un valor está dentro de un rango puede resultar útil en finanzas, probando que un salario es suficiente para alquilar o pedir una hipoteca, y también, para la localización, probando que algo o alguien está dentro de unos límites geográficos sin mostrar la localización exacta.

Protocolo 6.1. [13] Un probador, A, quiere demostrar a un verificador, B, que un valor m se encuentra entre $\{a, a+1, ..., b\}$.

a) *Preparación.* El centro de confianza realiza lo siguiente:

1.- Escoge dos primos p y q de modo que $(p-1)/2$ sea también un primo. Publica $n = pq$.

2.- Toma generadores de $\mathbb{Z}_p^*$ y $\mathbb{Z}_q^*$, g y h respectivamente.

Los valores n, g, h son públicos para el probador y el verificador.

El probador quiere demostrar que $m \in \{a, a+1, ..., b\}$, esto se cumple si y solo si $(m-a+1)(b-m+1) > 0$.

El probador, A, lleva a cabo los siguientes pasos:

3.- Escoge un entero aleatorio r y calcula $c = g^m h^r$ mód n.

4.- Escoge un entero aleatorio, w, distinto de 0 y calcula:

- $c_1 = \dfrac{c}{g^{a-1}}$ mód n.
- $c_2 = \dfrac{g^{b-1}}{c}$ mód n.
- Toma un entero aleatorio r' y obtiene $c' = c_1^{b-m+1} h^{r'}$ mód n.
- Toma un entero aleatorio r'' y obtiene $c'' = c'^{w^2} h^{r''}$ mód n.

5.- Escoge enteros no negativos $m_1, m_2, m_3, r_1, r_2, r_3$ de modo que cumplan los siguiente:

- $m_1 + m_2 + m_3 = w^2(m-a+1)(b-m+1)$.
- $r_1 + r_2 + r_3 = w^2((b-m+1)r + r') + r''$.

6.- Calcula:

- $c_1' = g^{m_1} h^{r_1}$ mód n.
- $c_2' = g^{m_2} h^{r_2}$ mód n.
- $c_3' = g^{m_3} h^{r_3}$ mód n.

7.- Tomando enteros aleatorios positivos s y t, y se calcula:

- $x = sm_1 + m_2 + m_3$.
- $y = m_1 + tm_2 + m_3$.
- $u = sr_1 + r_2 + r_3$.
- $v = r_1 + tr_2 + r_3$

b) *Envío de mensaje.* El probador envía al verificador la cadena:

$$(c, c_1, c_2, c', c'', c'_1, c'_2, c_1c'_3, x, y, u, v) \tag{6.1}$$

c) *Final del protocolo.* El verificador comprueba que:

9.- Comprueba:
- $c_1 = \dfrac{c}{g^{a-1}}$ mód n.
- $c_2 = \dfrac{g^{b+1}}{c}$ mód n.
- $c'' = c'_1c'_2c'_3$ mód n.

10.- Comprueba:
- $c'_1c'_2c'_3 = g^x h^u$ mód n.
- $c'_1c'_2c'_3 = g^y h^v$ mód n.

11.- Verifica que $x > 0$ e $y > 0$.

Si se superan con éxito los pasos 9, 10 y 11, el valor m se encuentra en el rango $\{a, ..., b\}$.

Observación 6.1. Veamos algunas observaciones sobre el protocolo anterior:

- Una opción más sencilla podría haber sido que el probador enviara x e y directamente al verificador y que éste comprobara que eran positivos. El problema que trae es que el probador puede escoger dos números aleatorios positivos y que el verificador aceptara, es por ello, que se recurren a otros valores que el verificador pueda comprobar.

- La elección de los enteros aleatorios $r', r'', w, r_1, r_2, r_3, m_1, m_2, m_3$ se hace para utilizarlos para ocultar los valores de los que son potencias. La seguridad se basa en la dificultad de resolver un logaritmo discreto.

- El uso de este tipo de pruebas no interactivas resulta más cómodo y seguro que una prueba interactiva. Ésto es debido a que muchas veces no es factible un intercambio de mensajes y por lo tanto, un solo mensaje y una comprobación resultan más eficientes.

¿Cuál es la utilidad de las pruebas de rango de conocimiento cero en el *blockchain*? Como hemos explicado, para registrar un nuevo bloque, éste debe pasar una serie de verificaciones o pruebas. Las pruebas de conocimiento cero permiten comprobar que la información que se va a añadir a un nuevo bloque es correcta sin poner en peligro dicha información. Por una parte, se realizarían pruebas de conocimiento cero de identificación y, por otra, dependiendo del contexto, una prueba de conocimiento cero de rango para determinar si el valor que se pretende añadir es correcto.

Conclusión

El objetivo principal de este trabajo es dar una idea general de las pruebas de conocimiento cero, desde sus orígenes como pruebas de membresía hasta una de sus aplicaciones actuales, pasando por los protocolos de identificación y de firma. Se ha hecho una revisión de la bibliografía más destacada sobre el tema y se ha estructurado cronológicamente en el trabajo.

Las pruebas de conocimiento cero permiten que el probador transmita al verificador que conoce una información sin revelarla.

Estas pruebas tienen un amplio futuro por delante puesto que mantener una información en secreto puede resultar muy útil. Hoy en día, es necesario seguir reinventándose en cuanto a seguridad, porque siempre puede haber impostores o trampas que pongan en peligro información delicada.

A lo largo de todo el trabajo hemos podido observar la evolución de la idea de conocimiento cero y como se ha ido adaptando según las necesidades. En un primer momento, se presenta como la demostración de la pertenencia de un elemento a un lenguaje a través de un intercambio de mensajes (Capítulos 2 y 3), siendo un concepto más teórico que práctico y con menos aplicaciones. Después, los conceptos teóricos se adaptan a la identificación y se crean los primeros protocolos de identificación de conocimiento cero (Capítulo 4), también interactivos. Posteriormente, se consiguen protocolos en los que no es necesario un intercambio de mensajes y se obtienen las pruebas no interactivas de conocimiento cero, que dan lugar a las firmas digitales de este tipo (Capítulo 5). Por último, se describe una nueva adaptación del concepto de prueba de conocimiento cero donde el probador demuestra al verificador que un valor pertenece a un rango sin mostrar dicho valor(Capítulo 6). Además, se ofrece una aplicación de este tipo de pruebas en la actualidad.

Uno de los problemas a los que me he enfrentado al realizar este trabajo ha sido la escasez y la disparidad de bibliografía, y que ésta estaba destinada a un público más técnico que teórico. En muchas ocasiones la labor del trabajo ha sido la de formalizar algunos conceptos que no resultaban del todo claros. Es el caso de una de las partes fundamentales de las pruebas de conocimiento cero, las máquinas de Turing probabilísticas, puesto que la bibliografía sobre este tema era casi inexistente y este trabajo pretendía dar una definición de prueba de conocimiento cero lo más formalizada posible. Otro problema que encontré era que la mayoría de protocolos de la bibliografía no estaban explicados de una manera clara y detallada, por lo que se tenían que analizar minuciosamente para encontrar las bases teóricas de los protocolos. A pesar de todo esto, he conseguido ampliar mi conocimiento sobre el tema que era el principal objetivo al realizar este trabajo. También he desarrollado capacidad de análisis y síntesis debido a la bibliografía dispar.

Bibliografía

[1] Babai, L. Graph isomorphism in quasipolynomial time. In *Proceedings of the forty-eighth annual ACM symposium on Theory of Computing* (2016), pp. 684–697.

[2] Bellare, M., and Goldwasser, S. New paradigms for digital signatures and message authentication based on non-interactive zero knowledge proofs. In *Advances in Cryptology—CRYPTO'89 Proceedings*.

[3] Blum, M., Feldman, P., and Micali, S. Non-interactive zero-knowledge and its applications. In *Proceedings of the twentieth annual ACM symposium on Theory of computing* (1988), pp. 103–112.

[4] Cecilia Pastorino. Blockchain: qué es, cómo funciona y cómo se está usando en el mercado. `https://www.welivesecurity.com/la-es/2018/09/04/blockchain-que-es-como-funciona-y-como-se-esta-usando-en-el-mercado/`.

[5] Encyclopedia of Mathematics. Probabilistic Turing Machine. `http://encyclopediaofmath.org/index.php?title=Probabilistic_Turing_machine&oldid=31204`.

[6] Encyclopedia of Mathematics. Turing Machine. `http://encyclopediaofmath.org/index.php?title=Turing_machine&oldid=31220`.

[7] Feige, U., Fiat, A., and Shamir, A. Zero-knowledge proofs of identity. *Journal of cryptology 1*, 2 (1988), 77–94.

[8] Fiat, A., and Shamir, A. How to prove yourself: Practical solutions to identification and signature problems. In *Conference on the theory and application of cryptographic techniques* (1986), Springer, pp. 186–194.

[9] Goldreich, O., Micali, S., and Wigderson, A. Proofs that yield nothing but their validity or all languages in np have zero-knowledge proof systems. *Journal of the ACM (JACM) 38*, 3 (1991), 690–728.

[10] Goldwasser, S., Micali, S., and Rackoff, C. The knowledge complexity of interactive proof-systems. In *Proceedings of the seventeenth annual ACM symposium on Theory of computing* (1985), pp. 291–304.

[11] GOLDWASSER, S., MICALI, S., and RACKOFF, C. The knowledge complexityof interactive proof systems. *SIAM J. COMPUT 18*, 1 (1989), 186–208.

[12] Jain, G. Zero knowledge proofs: A survey. *University of Pennsylvania* (2008).

[13] Koens, T., Ramaekers, C., and Van Wijk, C. Efficient zero-knowledge range proofs in ethereum.

[14] Menezes, A. J., Katz, J., Van Oorschot, P. C., and Vanstone, S. A. *Handbook of applied cryptography*. CRC press, 1996.

[15] Pass, R., and Shelat, A. A course in cryptograph. *Theoretical Foundation of Cryptography* (2010), 68–71.

[16] Schnorr, C.-P. Efficient signature generation by smart cards. *Journal of cryptology 4*, 3 (1991), 161–174.

[17] Shamir, A. Identity-based cryptosystems and signature schemes. In *Workshop on the theory and application of cryptographic techniques* (1984), Springer, pp. 47–53.

[18] Stinson, D. R., and Paterson, M. *Cryptography: theory and practice*. CRC press, 2018.