UNIVERSIDAD DE CHILE
FACULTAD DE CIENCIAS FÍSICAS Y MATEMÁTICAS
DEPARTAMENTO DE CIENCIAS DE LA COMPUTACIÓN

MECANISMO DE FILTRADO CRIPTOGRÁFICO PARA SISTEMAS DE COMUNICACIÓN ANÓNIMA BASADOS EN REENCRIPTACIÓN

MEMORIA PARA OPTAR AL TÍTULO DE INGENIERO CIVIL EN COMPUTACIÓN

JORGE ANDRÉS BAHAMONDE VEGA

PROFESOR GUÍA:
SR. ALEJANDRO HEVIA ANGULO

MIEMBROS DE LA COMISIÓN:
SR. JOHAN FABRY
SR. SERGIO OCHOA DE LORENZI

SANTIAGO DE CHILE
2013

Resumen de la Memoria para optar al Título de:
Ingeniero Civil en Computación
Por: Jorge Andrés Bahamonde Vega
Fecha: 21/11/2013
Profesor Guía: Alejandro Hevia Angulo

Mecanismo de filtrado criptográfico para sistemas de comunicación anónima basados en reencriptación

Las redes de comunicación anónima posibilitan nuevas formas de interacción entre las personas. En particular, la libre expresión de ideas contrarias a la opinión popular o la de la autoridad ha permitido, en ciertos casos, el inicio de procesos de cambio en las sociedades. Sin embargo, el anonimato puede ser abusado, lo cual lleva al cuestionamiento de la utilidad del mismo y a la tentación de no implementar estos sistemas. En este contexto, poder filtrar de manera segura mensajes según un criterio público permite resolver este problema, al permitir implementar sistemas de comunicación anónima donde es posible filtrar (descartar) mensajes que no calzan con una política pública predefinida.

Por ejemplo, el sistema de comunicación anónima pudiera establecer que sólo mensajes "certificados" (firmados) por entidades confiables pueden ser comunicados. En un posible escenario de uso de tal sistema, usuarios podrían "comprar" el acceso al sistema de comunicación anónima, vía una autorización de su autoridad certificadora local. La naturaleza distribuida de las implementaciones de canales de comunicación anónima aumenta la complejidad del problema - cada participante del sistema debe poder verificar si está procesando mensajes certificados o no.

En la presente memoria se trabajó en la definición e instanciación de un sistema de comunicación anónima que permitiese el filtrado de mensajes según una propiedad claramente definida. Para ello, primero se realizó un estudio de herramientas comúnmente aplicadas para obtener propiedades criptográficamente verificables, firmado y anonimato. A continuación, el trabajo se focalizó en el aspecto definicional.

Se logró construir una definición basada en conceptos criptográficos estándares. Luego, utilizando herramientas criptográficas analizadas, se construyó una instanciación de esta definición, utilizándose técnicas criptográficas formales y rigurosas para demostrar la seguridad del esquema.

La solución desarrollada no sólo logra resolver el problema planteado, sino que provee una base sobre la cual pueden realizarse múltiples mejoras y extensiones futuras.

Agradecimientos

Gracias a los que estuvieron conmigo. A mi familia. A Alejandro Hevia, mi profesor guía. A los que me dijeron que quizá era más fácil que lo que creía. A los que dijeron lo contrario. A los que me hicieron reír.

Bruno, Johan, Javiera, Gustavo, Pablo, Ricardo, Amanda, Patricio, Salomón, Diego, Ismael, Gabriel, Gonzalo, Daniel, Benjamín... Estoy seguro de que olvido a muchos. Todos, de alguna forma, ayudaron a esto. Todos, de alguna forma, son también parte de mi familia.

```
wow

    much thanks
```

Gracias a todos.

Tabla de contenido

1 Trasfondo y Antecedentes **1**
- 1.1 Notación general . . . 3
- 1.2 Conceptos matemáticos . . . 3
  - 1.2.1 Grupos algebraicos . . . 3
  - 1.2.2 Emparejamientos (pairings) bilineales . . . 4
  - 1.2.3 Supuestos en grupos algebraicos . . . 5
- 1.3 Conceptos criptográficos . . . 6
  - 1.3.1 Esquemas de encriptación . . . 6
  - 1.3.2 Realeatorización . . . 8
  - 1.3.3 Encriptación Homomórfica . . . 9
  - 1.3.4 Encriptación Umbral . . . 9
  - 1.3.5 Firmas digitales . . . 10
  - 1.3.6 Esquemas de commitment . . . 12
  - 1.3.7 Sistemas de demostración interactivos . . . 12
  - 1.3.8 Redes de mezcla . . . 20
- 1.4 Organización de esta Memoria . . . 23

2 Especificación del Problema **25**
- 2.1 Motivación . . . 25
- 2.2 Formalización . . . 26
- 2.3 Propiedades criptográficas . . . 27

3 Exploración de Soluciones **29**
- 3.1 Condiciones para una posible solución . . . 29
- 3.2 Firmas encriptadas verificables . . . 30
- 3.3 Esquemas de E-Cash . . . 31
  - 3.3.1 E-Cash compacto . . . 31
  - 3.3.2 E-Cash endosado . . . 34
  - 3.3.3 E-Cash transferible con un juez . . . 35
- 3.4 Sistema de demostración de Groth-Sahai . . . 35
- 3.5 Firmas sobre textos encriptados . . . 38
  - 3.5.1 Firmas conmutativas . . . 39
  - 3.5.2 Firmas sobre mensajes encriptados realeatorizables . . . 40

4 Descripción de la Solución **48**
- 4.1 Generación de claves . . . 49
- 4.2 Encriptación y autorización de mensajes . . . 49
- 4.3 Ejecución de la red . . . 50

5 Validación de la Solución **52**
5.1 Anonimato del esquema 52
5.2 Infalsificabilidad del esquema 62

6 Conclusiones **64**
6.1 Posibles mejoras y trabajo futuro 64
6.1.1 Instanciación de políticas de filtrado más complejas 65
6.1.2 Construcción de un esquema de firmas más eficiente 65
6.1.3 Uso de desencriptación distribuida 65
6.1.4 Utilización de propiedades de encriptación más sofisticadas 66
6.1.5 Demostración de seguridad en el Modelo de Composición Universal 66

Referencias **67**

Apéndices **72**
A Algoritmos del protocolo 72
A .1 Esquema de encriptación subyacente 72
A .2 Demostraciones utilizadas 75
B Demostraciones auxiliares 76
B .1 Seguridad semántica del esquema de encriptación subyacente 76
B .2 Infalsificabilidad del esquema de firma subyacente 82

Capítulo 1

Trasfondo y Antecedentes

La comunicación anónima (por ejemplo, a través de Internet) posee diversos usos, beneficios y desventajas. La posibilidad de establecer comunicación anónima entre partes permite, por ejemplo, la expresión de ideas contrarias a la opinión popular. Otros usos incluyen medios por los cuales comunidades pueden desahogar emociones y experiencias traumáticas, minimizar las diferencias de poder entre los participantes de la comunicación y permitir, en algunos casos, la resistencia de grupos minoritarios ante una mayoría más poderosa [25].

Existen diversos mecanismos a través de los cuales implementar redes de comunicación anónima. Uno de éstos es el de una red de mezcla, que utiliza un cierto número de servidores de mezcla que reciben mensajes encriptados de múltiples remitentes, permutándolos aleatoriamente y entregándolos al siguiente punto en la red de comunicación.

Una característica usual en estos sistemas es la robustez que alcanzan, bastando que uno de los servidores (de los m que pueden componer un camino en la red) realice el mezclado correcto para proveer de buenas garantías de seguridad a las partes involucradas. Concebidas por Chaum en 1981 [24], existen diversos tipos de redes de mezcla: redes de reencriptación y desencriptación y redes de sólo desencriptación. Las redes de reencriptación y desencriptación, por ejemplo, utilizan múltiples capas de encriptación que son removidas por cada servidor de mezcla.

Propuestas por Park, Itoh y Kurosawa [43] las redes de sólo reencriptación utilizan propiedades matemáticas para alterar la aleatoriedad de cada mensaje encriptado. Esto simplifica en parte el proceso, ya que los servidores de mezcla no requieren la existencia de claves de encriptación para cada servidor; adicionalmente, no es necesario conocer el camino que tomará el mensaje en la red antes de ingresarlo. Finalmente, posee mayor robustez que una red de desencriptación ante servidores de mezcla que se nieguen a participar en el proceso, no siendo necesario reiniciar la transmisión de los mensajes. Este trabajo se enfoca en este último tipo de red de comunicación anónima.

Un problema que ha cobrado relevancia en los sistemas de comunicación actuales es el de ofrecer anonimato y control de contenido a la vez. Esto, debido a que el anonimato ofrecido por un sistema de este tipo ha abierto las puertas a comportamientos ilícitos, como la adquisición de drogas ilegales [5], distribución de pornografía infantil [30] y otros.

Esto provoca no sólo un cuestionamiento de la utilidad de estos sistemas por parte del público general, sino que se ven involucrados en dificultades con la legislación. Un ejemplo particularmente reciente son los múltiples arrestos de personas involucradas con *Silk Road*, un mercado negro que operaba en la red de comunicación anónima Tor, en el cual se compraban y vendían drogas ilegales [49]. Eventualmente, estos inconvenientes llegan a causar el cuestionamiento del sentido mismo del anonimato [27]. De esta forma, organizaciones e individuos que aprecien los beneficios del anonimato pueden verse reticentes a utilizar o implementar uno de estos sistemas.

Una posibilidad para remediar esto es, ingenuamente, ejecutar el sistema de modo que los mensajes cuyo contenido sea inapropiado pueden ser eliminados luego de que hayan sido procesados por el sistema (es decir, antes de su publicación). Esto es claramente insuficiente, ya que eventualmente podría gastarse una parte considerable del ancho de banda del sistema de comunicación en mensajes que serán finalmente borrados. Por otro lado, el filtrado al momento mismo del ingreso (antes de ocultar su contenido) violaría el anonimato del sistema.

Adicionalmente, si los mensajes se encuentran encriptados, no resulta evidente el mecanismo a través del cual realizar un filtrado (de acuerdo a alguna política acordada) sin revelar información de los mensajes originales o violar las propiedades del esquema de encriptación.

Además de esto, la posibilidad de realizar filtrado privado en una red de comunicación anónima abre la puerta a otras aplicaciones. Por ejemplo, un sistema de comunicación anónima que implemente un control de acceso en alguna manera. Si bien podría pensarse que esquemas de dinero electrónico pueden ser una solución adecuada a esta problemática (siendo el pago el control de acceso implementado), estos sistemas de dinero electrónico se ocupan del anonimato y/o de que las transacción sea justa o segura. Así, no es claro cómo combinarlos para adjuntar algún mensaje que se desee publicar de manera anónima.

Desde el punto de vista teórico, el problema a tratar resulta interesante, pues se busca conjugar diversos aspectos de la criptografía para lograr un esquema que cumpla con las propiedades deseadas: es necesario asegurar que al red de comunicación realice el filtro de forma correcta y verificable de los mensajes. El sistema, además, debe preservar el anonimato de la red de comunicación, al menos, de los participantes honestos. La conservación de estas propiedades de manera simultánea representa el principal desafío del problema.

En el presente capítulo, se presenta un resumen de los conceptos matemáticos y criptográficos necesarios para comprender las definiciones y demostraciones posteriores. Para comenzar, se especificarán conceptos matemáticos involucrados en la construcción de los diferentes esquemas y mecanismos criptográficos que se analizarán.

1.1 Notación general

Se utilizarán MAYÚSCULAS PEQUEÑAS para nombres de funciones, algoritmos y protocolos. Por otro lado, se utilizarán cursivas ($\mathscr{A}, \mathscr{B}$...) para adversarios y entidades. Se usará fuente negrita de pizarra ($\mathbb{A}$, $\mathbb{B}$...) para grupos algebraicos. Finalmente, se utilizará fuente monoespaciada para valores que expresan conceptos en lenguaje natural (como `Ok` o `Error`).

Se denotará $a \xleftarrow{\$} B$ en dos situaciones. Si B es un conjunto, a es un elemento escogido de manera uniformemente aleatoria de éste. Si B es un algoritmo o protocolo, a es la salida de éste, siendo B un algoritmo o protocolo aleatorizado. El caso $a \leftarrow B$ representa la salida de un algoritmo u operación.

1.2 Conceptos matemáticos

La siguiente definición será utilizada en lo sucesivo, particularmente en las definiciones de seguridad de esquemas criptográficos.

Definición 1 (Función despreciable)**.** *Se dice que una función f es despreciable si para todo polinomio p existe un entero positivo N tal que para todo entero $n > N$, $f(n) < \frac{1}{p(n)}$.*

1.2.1 Grupos algebraicos

Dado un grupo algebraico $(\mathbb{G}, \cdot)$, se denota a^{-1} al inverso de a, y $1_{\mathbb{G}}$ al elemento neutro de la operación $\cdot$. Adicionalmente, se denomina $\mathbb{Z}_p$ al conjunto de los enteros positivos menores que p, y $\mathbb{Z}_p^*$ al conjunto de enteros en $\mathbb{Z}_p$ coprimos con p. Finalmente, se utilizará $\mathbb{G}$ tanto para el grupo en sí como para el conjunto, de modo de ahorrar en notación.

Se define a^n con $a \in \mathbb{G}, n \in \mathbb{Z}$ como el producto de n copias de a, definiéndose $a^0 \equiv 1_{\mathbb{G}}$. En el caso $n < 0$, $a^n \equiv (a^{-n})^{-1} = (a^{-1})^{(-n)}$.

Este estilo de notación es denominada en la literatura como *notación multiplicativa*.

Definición 2 (Orden de un elemento)**.** *Dado un elemento a de un grupo $\mathbb{G}$, se define su orden como el entero positivo m más pequeño tal que $a^m = 1_{\mathbb{G}}$. Si no existiera tal entero, se dirá que a tiene orden infinito. Se denota como $ord(a)$.*

Definición 3 (Orden de un grupo)**.** *El orden de un grupo se define como su cardinalidad: es decir, el número de elementos en el conjunto $\mathbb{G}$. Se denota como $|\mathbb{G}|$.*

Definición 4 (Conjunto generador de un grupo)**.** *Se dice que un conjunto S es un generador de un grupo* $\mathbb{G}$ *si todo elemento de* $\mathbb{G}$ *puede ser expresado como una combinación (utilizando* $\cdot$*) de una cantidad finita de elementos (repetibles) de S y sus inversos.*

Definición 5 (Grupo cíclico)**.** *Un grupo cíclico es aquel que posee un conjunto generador con sólo un elemento. Formalmente, un grupo* $\mathbb{G}$ *es cíclico si se cumple:*

$$\exists a \in \mathbb{G}\ \forall x \in \mathbb{G}\ \exists n \in \mathbb{Z},\ a^n = x$$

Se dice que a es un ***generador*** *de* $\mathbb{G}$*.*

Los siguientes teoremas serán de utilidad más adelante, encontrándose sus demostraciones en [46, §6.5]:

Teorema 1. *Todo grupo* $\mathbb{G}$*, tal que* $|\mathbb{G}|$ *es primo, es cíclico. Adicionalmente, todo elemento* $a \in \mathbb{G}$ *tal que* $a \neq 1_{\mathbb{G}}$ *es generador.*

Teorema 2. *En un grupo finito* $\mathbb{G}$ *de orden p, si g es un generador, se cumple:*

$$\forall a \in \mathbb{G}\ \exists n \in \mathbb{Z}_p^*\ \ g^n = a$$

1.2.2 Emparejamientos (pairings) bilineales

Los *pairings* son construcciones matemáticas que, en los últimos años, han posibilitado la creación de esquemas criptográficos con propiedades extremadamente interesantes. A continuación se presentan su definición y sus propiedades [33] [42]:

Definición 6 (Pairing bilineal)**.** *Dados* $\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T$ *grupos cíclicos de orden primo p, una función* $e: \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ *es un pairing bilineal si cumple:*

$$\forall a \in \mathbb{G}_1, b \in \mathbb{G}_2, m, n \in \mathbb{Z}_p^*,\ \ e(a^m, b^n) = e(a,b)^{mn}$$

Adicionalmente, se dice que un pairing es ***no degenerado*** *si, dado* $a \neq 1_{\mathbb{G}_1} \in \mathbb{G}_1$*, existe* $b \in \mathbb{G}_2$ *tal que* $e(a,b) \neq 1_{\mathbb{G}_T}$*.*

Los *pairings* son clasificados en la literatura en tres categorías básicas, de acuerdo a las relaciones entre $\mathbb{G}_1$ y $\mathbb{G}_2$:

Tipo 1: $\mathbb{G}_1 = \mathbb{G}_2$

Tipo 2: $\mathbb{G}_1 \neq \mathbb{G}_2$, pero se tiene un homomorfismo computable $\phi: \mathbb{G}_2 \to \mathbb{G}_1$. Sin embargo, no se tiene un homomorfismo computable $\phi': \mathbb{G}_1 \to \mathbb{G}_2$.

Tipo 3: $\mathbb{G}_1 \neq \mathbb{G}_2$, y no se tienen homomorfismos computables entre $\mathbb{G}_1$ y $\mathbb{G}_2$.

El caso en que existan homomorfismos en ambas direcciones puede reinterpretarse como un pairing del tipo 1.

1.2.3 Supuestos en grupos algebraicos

A continuación se presentan algunos supuestos sobre grupos algebraicos, útiles al momento de demostrar la seguridad de algunos esquemas criptográficos.

Supuesto Diffie-Hellman decisional

El supuesto Diffie-Hellman decisional [12] se cumple si no existe un algoritmo (posiblemente aleatorizado) que, en un tiempo polinomial, resuelva el siguiente problema con una probabilidad no despreciable:

Definición 7 (Problema de Diffie-Hellman decisional)**.** *Considérese un grupo cíclico $\mathbb{G}$ de orden p. Dado $g \in \mathbb{G}$, se pide diferenciar entre las siguientes tuplas:*

$$(g^a, g^b, g^{ab}), \text{ con } a, b \xleftarrow{\$} \mathbb{Z}_p$$
$$(g^a, g^b, g^c), \text{ con } a, b, c \xleftarrow{\$} \mathbb{Z}_p$$

Supuesto Diffie-Hellman computacional

El supuesto Diffie-Hellman computacional [12] dice que ningún algoritmo (posiblemente aleatorizado) puede resolver el siguiente problema en tiempo polinomial con una probabilidad no despreciable:

Definición 8 (Problema de Diffie-Hellman computacional)**.** *Considérese un grupo cíclico $\mathbb{G}$ de orden p. Dada la tupla (g^a, g^b), con $a, b \xleftarrow{\$} \mathbb{Z}_p$, se pide calcular el valor de g^{ab}.*

Supuesto Lineal Decisional

El supuesto lineal decisional [14] asegura que ningún algoritmo (posiblemente aleatorizado) puede resolver el siguiente problema en tiempo polinomial con probabilidad no despreciable:

Definición 9 (Problema Lineal Decisional)**.** *Considérese un grupo cíclico* $\mathbb{G}$*. Dados* $u,v,h \in \mathbb{G}$*, donde* $u,v,h \xleftarrow{\$} \mathbb{G}$*, se pide diferenciar entre las siguientes tuplas:*

$$u,v,h,u^a,v^b,h^{a+b}, \text{ con } a,b \xleftarrow{\$} \mathbb{Z}_p$$
$$u,v,h,u^a,v^b,h^c, \text{ con } a,b,c \xleftarrow{\$} \mathbb{Z}_p$$

1.3 Conceptos criptográficos

A continuación se presentan algunos conceptos criptográficos básicos. Se siguen las definiciones expuestas en [40], pero con la notación utilizada en esta memoria.

1.3.1 Esquemas de encriptación [40, cap. 10]

Definición 10 (Encriptación de clave pública)**.** *Un* ***esquema de encriptación de clave pública*** *es una tupla de algoritmos probabilísticos de tiempo polinomial* $(\text{KEYGEN}, \text{ENCRYPT}, \text{DECRYPT})$ *que satisface:*

- KEYGEN *recibe como entrada un parámetro de seguridad k, y entrega como salida un par de claves* (pk,dk)*. La primera se denomina* ***clave pública o de encriptación****, y la segunda como* ***clave privada o de desencriptación****.*
- ENCRYPT *recibe como entrada una clave pública pk y un* ***mensaje o texto plano*** *M de un espacio de mensajes subyacente, posiblemente dependiente de pk. Emite como salida un* ***texto cifrado*** *c. Se denota esta operación como* $c \xleftarrow{\$} \text{ENCRYPT}_{pk}(M)$
- DECRYPT *recibe una clave privada sk y un texto cifrado c, y emite un mensaje M. Se supone, sin pérdida de generalidad, que* DECRYPT *es un algoritmo determinista* [1]*. Finalmente, la notación usual es* $M \leftarrow \text{DECRYPT}_{dk}(c)$.

[1] Puede definirse, para un sistema de encriptación donde DECRYPT utiliza aleatoriedad r, un esquema donde esta aleatoriedad es escogida al momento de la encriptación y se adjunta al texto cifrado c, formando $c' = (c,r)$; para desencriptar, se utiliza esta aleatoriedad, volviéndose la desencriptación un proceso determinístico.

- *Para todo k,* $(pk, sk) \xleftarrow{\$} \text{KEYGEN}(k)$*, y todo mensaje M en el espacio subyacente apropiado, se cumple que*

$$\text{DECRYPT}_{dk}(\text{ENCRYPT}_{pk}(M)) = M$$

La clave pública se distribuye, haciéndose públicamente disponible para cualquiera que desee encriptar un mensaje.

Seguridad de un esquema de encriptación

Para definir la seguridad de un esquema de encriptación de clave pública, es útil definir el siguiente experimento:

Experimento 1 (Experimento de confidencialidad)**.** $\text{PubK}^{\text{eav}}_{\mathscr{A},\Pi}(k)$*:*

1. *Se ejecuta* $\text{KEYGEN}(k)$ *y se obtienen claves* (pk, dk).
2. *Se elige un bit al azar* $b \xleftarrow{\$} \{0,1\}$.
3. *Se entrega pk al adversario* $\mathscr{A}$*, que emite un par de mensajes* m_0, m_1*, con* $|m_0| = |m_1|$ *y pertenecientes al espacio de mensajes asociado a pk.*
4. *Se calcula* $c \xleftarrow{\$} \text{ENCRYPT}_{pk}(m_b)$*, que se entrega a* $\mathscr{A}$*. Esta operación se denota como* $c \xleftarrow{\$} \text{LOR}(m_0, m_1)$.
5. $\mathscr{A}$ *emite un bit* b'.
6. *La salida del experimento se define como 1 si* $b = b'$*, y 0 si no.*

Sin pérdida de generalidad, un adversario para este experimento puede descomponerse como $\mathscr{A} = (\mathscr{A}_1, \mathscr{A}_2)$, donde $\mathscr{A}_1$ recibe pk, y retorna m_0, m_1 al experimento PubK^{eav}; $\mathscr{A}_2$ recibe el texto cifrado c por parte del experimento y emite el bit de adivinanza b'. Existe un estado que se transfiere entre estas dos componentes de $\mathscr{A}$, que en general no se denotará. Con esto, se define la indistinguibilidad ante ataques de texto plano escogido de la siguiente forma:

Definición 11 (Indistinguibilidad ante ataques de texto plano escogido (IND-CPA))**.** *Un esquema de encriptación de clave pública* $\Pi = (\text{KEYGEN}, \text{ENCRYPT}, \text{DECRYPT})$ *es* ***indistinguible ante ataques de texto plano escogido*** *si para todo adversario probabilístico de tiempo polinomial* $\mathscr{A}$*, existe una función despreciable* f *tal que:*

$$\mathbb{P}[\text{PubK}^{\text{eav}}_{\mathscr{A},\Pi}(k) = 1] \leq \frac{1}{2} + f(k)$$

Se define, así, la ventaja IND-CPA (del inglés indistinguishability against chosen-ciphertext attacks) de $\mathscr{A}$:

$$\mathrm{Adv}_{\Pi}^{\text{IND-CPA}}(\mathscr{A}) = |2 \cdot \mathbb{P}[\mathrm{PubK}_{\mathscr{A},\Pi}^{\text{eav}}(k) = 1] - 1|$$

De esta forma, un esquema de encriptación de clave pública será indistinguible ante ataques de texto plano escogido si para todo adversario, su ventaja IND-CPA es despreciable.

Otro nombre para esta propiedad es el de *seguridad semántica* [35].

1.3.2 Realeatorización

Diversos esquemas de encriptación poseen la particularidad de ser *realeatorizables*. La operación de realeatorización (a veces denominada como reencriptación) permite obtener un texto cifrado c' a partir de un texto cifrado c, de modo que ambos corresponden al mismo mensaje, pese a que los textos cifrados sean distintos (con alta probabilidad). Más aún, un texto cifrado realeatorizado es indistinguible de un texto cifrado "original". Formalmente [20]:

Experimento 2 (Experimento de realeatorización)**.** $\mathrm{ReRand}_{\mathscr{A},\Pi,\text{RANDOM}}(k)$*:*

1. *Se ejecuta* $\text{KEYGEN}(k)$ *y se obtienen claves* (pk, dk).
2. *Se entrega pk al adversario* $\mathscr{A}$*, que emite un texto cifrado c, perteneciente al espacio de textos cifrados válidos.*
3. *Se elige un bit al azar* $b \xleftarrow{\$} \{0,1\}$*, calculándose:*

$$c' \xleftarrow{\$} \begin{cases} \text{ENCRYPT}_{pk}(\text{DECRYPT}_{dk}(c)) & \text{si } b = 0 \\ \text{RANDOM}_{pk}(c) & \text{si } b = 1 \end{cases}$$

4. *Se entrega* c' *a* $\mathscr{A}$*, que emite un bit* b'.
5. *La salida del experimento se define como 1 si* $b = b'$*, y 0 si no.*

Con esto, se define realeatorización como sigue:

Definición 12 (Realeatorización)**.** *Un esquema de encriptación de clave pública* $\Pi = (\text{KEYGEN}, \text{ENCRYPT}, \text{DECRYPT})$ *es* ***realeatorizable*** *si existe una función* RANDOM *de modo que, para todo adversario probabilístico de tiempo polinomial* $\mathscr{A}$ *exista una función despreciable* f *tal que:*

$$\mathbb{P}[\mathrm{ReRand}_{\mathscr{A},\Pi,\text{RANDOM}}(k) = 1] \leq \frac{1}{2} + f(k)$$

1.3.3 Encriptación Homomórfica [40, cap. 11]

Una propiedad útil de ciertos esquemas de encriptación de clave pública es el de ser esquemas *homomórficos* (u homeomórficos), propiedad que permite crear textos cifrados a partir de otros:

Definición 13 (Esquema de encriptación homomórfica de clave pública)**.** *Un esquema de encriptación de clave pública* (SETUP, ENCRYPT, DECRYPT) *es* ***homomórfico*** *si, para todo k, para todo* (pk, dk) *emitidos por* SETUP(k)*, es posible definir grupos algebraicos* $(\mathbb{C}, \cdot)$*,* $(\mathbb{M}, \circ)$ *de modo que:*

- *El espacio de mensajes es* $\mathbb{M}$*, y todos los textos cifrados emitidos por* ENCRYPT$_{pk}$ *son elementos de* $\mathbb{C}$*.*
- $\forall M_1, M_2 \in \mathbb{M}, c_1, c_2 \in \mathbb{C}$*, tales que* $M_1 = \text{DECRYPT}_{dk}(c_1)$ *y* $M_2 = \text{DECRYPT}_{dk}(c_2)$*, se cumple*

$$\text{DECRYPT}_{dk}(c_1 \cdot c_2) = M_1 \circ M_2,$$

1.3.4 Encriptación Umbral [13]

Los esquemas de encriptación umbral corresponden a una variante de esquemas de encriptación en la que, mientras que la encriptación es realizada por una sola entidad, se requiere de la colaboración de múltiples participantes para realizar la desencriptación. Se denota como un esquema de encriptación umbral (t, n) a un esquema en el que cualquier subconjunto de t o más entidades de un conjunto de n puede realizar la desencriptación de un texto cifrado, mientras que cualquier conjunto de menos de t participantes no puede obtener información alguna de éste.

Más formalmente, se define un esquema de encriptación umbral como sigue:

Definición 14 (Esquema de encriptación umbral (t, n))**.** *Un esquema de encriptación umbral (t, n) es una tupla de algoritmos probabilísticos de tiempo polinomial compuesta por* KEYGEN*,* ENCRYPT*,* SHAREDECRYPT*,* SHAREVERIFY *y* COMBINE*, que satisface:*

- KEYGEN *recibe como entrada el número de servidores de desencriptación n, un umbral* $t \in \{1, ..., n\}$ *y un parámetro de seguridad k. Emite una tupla* (pk, vk, dk)*, donde pk se denomina la* ***clave pública****, vk se denomina la* ***clave de verificación****, y* $dk = \{dk_1, ...dk_n\}$ *es un vector de n* ***trozos de clave privada****. El servidor de desencriptación i recibe* (i, dk_i)*.*
- ENCRYPT*, de manera análoga a un esquema de encriptación, recibe una clave pública y un mensaje, entregando un texto cifrado.*
- SHAREDECRYPT *recibe como entrada la clave pública pk, un texto cifrado c y uno de los n trozos de clave privada de sk. Emite un trozo de desencriptación* $\mu = (i, \hat{\mu})$ *del texto cifrado, o* $\perp$ *(donde* $\perp$ *simboliza un error).*

- SHAREVERIFY *recibe como entrada la clave pública pk, la clave de verificación vk, un texto cifrado c y un trozo de desencriptación* $(i, \hat{\mu})$. *Emite un bit b, con* $b = 1$ *significando "Válido" y* $b = 0$ *significando "Inválido".*

- COMBINE *recibe pk, vk, un texto cifrado c y t trozos de desencriptación* $\{\mu_1, ..., \mu_t\}$. *Emite un mensaje M o* $\perp$.

- *Para todas las claves pk, vk,* sk_i *emitidas por* KEYGEN*, y para todo mensaje M de modo que* $C \xleftarrow{\$} \text{ENCRYPT}_{pk}(M)$*, se cumple:*

$$\text{SHAREVERIFY}_{pk,vk}(C, \text{SHAREDECRYPT}_{pk}(i, sk_i, C)) = 1$$

- *Para todas las claves pk, vk,* sk_i *emitidas por* KEYGEN*, y para todo mensaje M de modo que* $C \xleftarrow{\$} \text{ENCRYPT}_{pk}(M)$*, si* $S = \{\mu_1, ...\mu_t\}$ *es un conjunto de trozos de desencriptación* $\mu_i \leftarrow \text{SHAREDECRYPT}_{pk}(i, sk_i, C)$ *para t trozos de clave privada distintas del conjunto sk, se tiene:*

$$\text{COMBINE}_{pk,vk}(C, S) = M$$

La seguridad de un esquema de encriptación umbral puede definirse de forma análoga al de un esquema de encriptación usual. Adicionalmente, se puede requerir que los trozos de desencriptación no revelen información sobre los trozos de clave privada, incluso si los dueños de estos trozos son maliciosos; de igual forma, se requiere que estos participantes maliciosos no puedan recuperar un texto plano a partir de un número insuficiente de participantes que colaboren en la desencriptación.

Una manera de formalizar estos requerimientos es a través del concepto de *simulabilidad de trozos*, detallado en [21, §2.3]. Es necesario notar, sin embargo, que no es la única forma de caracterizar la seguridad de un esquema de encriptación umbral, existiendo variaciones en la literatura.

1.3.5 Firmas digitales [40, cap. 12]

Definición 15 (Esquema de firmas)**.** *Un* ***esquema de firmas*** *es una tupla de algoritmos probabilísticos de tiempo polinomial* (KEYGEN, SIGN, VERIFY) *que satisface:*

- KEYGEN *recibe como entrada un parámetro de seguridad k, entregando como salida un par de claves* (vk, sk). *La primera se denomina* ***clave pública o de verificación****, y la segunda como la* ***clave privada o de firmado****.*

- SIGN *recibe como entrada una clave privada sk y un* ***mensaje*** *M de un espacio de mensajes subyacente, posiblemente dependiente de vk. Emite como salida una* ***firma*** σ. *Se denota esta operación como* $\sigma \xleftarrow{\$} \text{SIGN}_{sk}(M)$

- VERIFY *recibe una clave pública vk, un mensaje M y una firma* σ*. Emite un bit b, con* $b = 1$ *significando "Válido" y* $b = 0$ *significando "Inválido".* VERIFY *es un algoritmo determinista; esto se escribe* $b \leftarrow \text{VERIFY}_{vk}(M, \sigma)$.

- *Para todo k,* $(vk, sk) \xleftarrow{\$} \text{KEYGEN}(k)$*, y todo mensaje M en el espacio subyacente apropiado, se cumple que:*

$$\text{VERIFY}_{vk}(M, \text{SIGN}_{sk}(M)) = 1$$

Se denota que σ es una *firma válida* sobre un mensaje M (con la clave pública *vk* implícita en el contexto) si $\text{VERIFY}_{vk}(M, \sigma) = 1$.

Seguridad de un esquema de firmas

Para definir la seguridad de un esquema de firmas, es útil considerar el siguiente experimento, dado un esquema de firmas $\Pi = (\text{KEYGEN}, \text{SIGN}, \text{VERIFY})$:

Experimento 3 (Experimento de infalsificabilidad)**.** $\text{Sig-forge}^{\text{cma}}_{\mathscr{A},\Pi}(k)$*:*

1. *Se ejecuta* $\text{KEYGEN}(k)$*, obteniéndose claves* (vk, sk).
2. *El adversario* $\mathscr{A}$ *recibe pk y acceso a un oráculo* $\text{SIGN}_{sk}(\cdot)$*. Este oráculo emite una firma* $\text{SIGN}_{sk}(M)$ *al recibir un mensaje M elegido por* $\mathscr{A}$*. Luego* $\mathscr{A}$ *emite* (M, σ).
3. *Sea SC el conjunto de pares (mensaje, firma) cuyas firmas fueron entregadas por* SIGN *al adversario* $\mathscr{A}$ *durante su ejecución. La salida del experimento es 1 si* $(M, \sigma) \notin SC$ *y* $\text{VERIFY}_{vk}(M, \sigma) = 1$.

En el contexto de esquemas criptográficos, se dice que un adversario tiene acceso a un oráculo cuando éste puede realizar invocaciones a esta funcionalidad sin ser capaz de observar su funcionamiento interno. Por ejemplo, el acceso a un oráculo $\text{SIGN}_{sk}(\cdot)$ permite que el usuario entregue un mensaje a este oráculo, obteniendo una firma sobre éste. Sin embargo, el adversario no puede observar ni alterar el funcionamiento del mecanismo de firma involucrado: sólo es capaz de observar las entradas y las salidas. Con este experimento, se construye el concepto de infalsificabilidad ante ataques de mensaje escogido de la siguiente forma:

Definición 16 (Infalsificabilidad ante ataques de mensaje escogido (UF-CMA))**.** *Un esquema de firmas* $\Pi = (\text{KEYGEN}, \text{SIGN}, \text{VERIFY})$ *es* ***existencialmente infalsificable ante un ataque de mensaje escogido*** *si para todo adversario probabilístico de tiempo polinomial* $\mathscr{A}$*, existe una función despreciable* f *tal que*

$$\mathbb{P}[\text{Sig-forge}^{\text{cma}}_{\mathscr{A},\Pi}(k) = 1] \leq f(k)$$

Se denomina a esta probabilidad como la ventaja UF-CMA ($\text{Adv}^{\text{uf-cma}}$*) de* $\mathscr{A}$.

1.3.6 Esquemas de commitment

Un esquema de *commitment* es una herramienta criptográfica que permite la transferencia segura de un valor entre dos partes, sin revelarse su contenido hasta un momento posterior. El concepto de *commitment* fue concebido en 1981 por Manuel Blum [10]. Más formalmente, se puede definir un esquema de commitment de la siguiente forma [26]:

Definición 17 (Esquema de commitment)**.** *Un esquema de commitment* $(\text{SETUP}, \mathscr{S}, \mathscr{R})$ *es un protocolo de dos fases entre dos entidades probabilísticas de tiempo polinomial* $\mathscr{S}$ *y* $\mathscr{R}$*, llamados emisor y receptor respectivamente, tal que se cumple:*

- *En la primera fase (fase de* ***commitment****), dado* $\sigma \xleftarrow{\$} \text{SETUP}$*, el enviador* $\mathscr{S}$ *se compromete con un valor p computando un par de claves* (com, dec) *y enviando com, la clave de commitment, a* $\mathscr{R}$*.*
- *En la segunda fase, (fase de* ***decommitment****),* $\mathscr{S}$ *muestra p y dec a* $\mathscr{R}$*; el receptor* $\mathscr{R}$ *puede verificar que dec es válida y convencerse de que p es efectivamente el valor usado por* $\mathscr{S}$ *en la primera fase.*

Un esquema de commitment posee *ocultación* computacional si, durante la primera fase, un adversario probabilístico de tiempo polinomial en el papel de receptor no puede obtener información alguna del valor p contenido en com si el emisor es honesto; por otro lado, un esquema de commitments es computacionalmente *vinculante* si un adversario probabilístico de tiempo polinomial en el papel de emisor no puede convencer a un receptor honesto, en la segunda fase, de que el commitment enviado contiene un valor distinto al utilizado en la primera fase. Se requiere, en general, que un esquema de commitments posea estas dos propiedades.

Estas propiedades se denominan *ocultación perfecta* (u ocultación incondicional) y *perfectamente vinculante* (o incondicionalmente vinculante), respectivamente, si las definiciones se mantienen ante adversarios con poder computacional ilimitado; sin embargo, ambas propiedades no pueden tenerse de forma simultánea [28, §4.3]. En lo sucesivo, se denotará $\langle x \rangle$ a un commitment que contiene el valor x.

1.3.7 Sistemas de demostración interactivos

Los sistemas de demostración consisten en protocolos que permiten mostrar que ciertos valores cumplen una cierta relación, sin revelar más información que la disponible de forma pública. Para ejemplificar, un sistema de demostración permitiría que $\mathscr{A}$ demuestre a $\mathscr{B}$ que un cierto grafo G posee un camino hamiltoniano, pero sin revelar ninguna información de este camino. El camino,

por ejemplo, podría ser eventualmente entregado en forma de commitment, lo que podría permitir operar con él sin conocerlo.

En este ejemplo, se denomina *instancia* al grafo G (en el sentido de una instancia del problema de camino hamiltoniano), y *testigo* al camino en sí (que no se desea revelar). En las sucesivas definiciones de esta sección se utilizará x para las instancias y w para los testigos. Se llama L al conjunto de instancias que denota la posesión de una propiedad en particular (en este caso, "poseer un camino hamiltoniano"), y R a la relación que representa que w es un testigo válido para x, para el conjunto L (en este caso, "w es un camino hamiltoniano en x"). Se denota como P (*prover*, el demostrador) a la parte que desea demostrar que $x \in L$, y V (*verifier*, el verificador) a la parte que verifica este hecho.

Así, un sistema de demostración es un trío de algoritmos $(\text{SETUP}, \text{P}, \text{V})$, donde P y V pueden ser interactivos. En adición a P y V, SETUP emite parámetros globales, denotados como ω. Este parámetro global se denomina *cadena de referencia común* en el modelo de seguridad que lleva el mismo nombre.

En el caso en que P y V sean algoritmos de una ronda (es decir, P y V son algoritmos aleatorizados donde una de las entradas de V es la salida de P), se dice que el sistema de demostración es *no interactivo*. Para el caso de sistemas interactivos, se denotará $out(\text{P}, \text{V})$ como la salida de la interacción entre P y V (1 para aceptación o 0 para rechazo). Adicionalmente, se denotará $tr \leftarrow \langle \text{P}, \text{V} \rangle$ como el conjunto de mensajes intercambiados entre las partes, también denominado la *traza* de la interacción.

Más formalmente, y como se explica en [3, cap. 9] y [20], se definen a continuación diversas propiedades para sistemas de demostración tanto interactivos como no interactivos.

Completitud y consistencia

La propiedad de completitud encapsula la idea de que un demostrador honesto debe poder realizar la demostración de manera exitosa:

Definición 18 (Completitud)**.** *Un sistema de demostración* $(\text{SETUP}, \text{P}, \text{V})$ *es* ***completo*** *cuando,* $\forall \omega_{crs} \xleftarrow{\$} \text{SETUP}(k), \forall (x,w) \in R, \exists \text{P}:$

$$\mathbb{P}[out(\text{P}(\omega_{crs}, x, w), \text{V}(\omega_{crs}, x)) = 1] \geq 2/3$$

Por otro lado, un sistema de demostración no interactivo $(\text{SETUP}, \text{P}, \text{V})$ *es* ***completo*** *cuando,* $\forall \omega_{crs} \xleftarrow{\$} \text{SETUP}(k), \forall (x,w) \in R, \forall \pi \xleftarrow{\$} \text{P}(\omega_{crs}, x, w):$

$$\mathbb{P}[\text{V}(\omega_{crs}, x, \pi) = 1] = 1$$

La propiedad de consistencia representa la idea de que un verificador honesto no debiera poder ser engañado por un demostrador malicioso que intente demostrar un hecho falso:

Definición 19 (Consistencia)**.** *Un sistema de demostración* $(\text{SETUP}, \text{P}, \text{V})$ *es* ***consistente*** *cuando, para todo adversario probabilístico de tiempo polinomial* $\mathscr{A}$*, se cumple:*
$\forall \omega_{crs} \xleftarrow{\$} \text{SETUP}(k), \forall x \notin L, \forall \text{P}^* :$

$$\mathbb{P}[out(\text{P}(\omega_{crs}, x, w), \text{V}(\omega_{crs}, x)) = 1] \leq 1/3$$

Por otro lado, un sistema de demostración no interactivo $(\text{SETUP}, \text{P}, \text{V})$ *es* ***consistente*** *cuando, para todo adversario probabilístico de tiempo polinomial* $\mathscr{A}$*, se cumple:*
$\forall \omega_{crs} \xleftarrow{\$} \text{SETUP}(k) :$

$$((x, \pi) \xleftarrow{\$} \mathscr{A}(\omega_{crs}) \wedge x \notin L) \implies \mathbb{P}[\text{V}(\omega_{crs}, x, \pi) = 1] \leq f(k)$$

para una función f *despreciable. En el caso en que esta probabilidad sea exactamente 0, la propiedad se denomina* ***consistencia perfecta****.*

Emulación extendida con testigo y extraibilidad

Parte de la literatura utiliza el concepto de *argumento* en vez de demostración cuando el esquema no posee consistencia perfecta. Adicionalmente, las siguientes dos definiciones refuerzan el concepto de consistencia. Para sistemas interactivos, se tiene la *emulación extendida con testigo*:

Definición 20 (Emulación extendida con testigo)**.** *Un sistema de demostración* $(\text{SETUP}, \text{P}, \text{V})$ *posee* ***emulación extendida con testigo*** *si para todo algoritmo determinístico de tiempo polinomial* P *existe un algoritmo de tiempo esperado polinomial* E*, que emula la interacción entre* P *y* V*, tal que para todo adversario de tiempo polinomial* $\mathscr{A}' = (\mathscr{A}, \text{P}^*)$*, se cumple:*

$$\begin{aligned}
&\mathbb{P}[\omega \xleftarrow{\$} \text{SETUP}(k); (x, s) \xleftarrow{\$} \mathscr{A}(\omega); tr \leftarrow \langle \text{P}^*(\omega, x, s), \text{V}(\omega, x) \rangle : \mathscr{A}(tr) = 1] \\
\approx\ &\mathbb{P}[\omega \xleftarrow{\$} \text{SETUP}(k); (x, s) \xleftarrow{\$} \mathscr{A}(\omega); (tr, w) \leftarrow E^{\langle \text{P}^*(\omega, x, s), \text{V}(\omega, x) \rangle} : \mathscr{A}(tr) = 1 \\
&\qquad \text{y si } tr \text{ corresponde a una interacción exitosa, entonces } (x, w) \in R]
\end{aligned}$$

En esta definición, s representa el estado del demostrador malicioso P^*, incluyendo la aleatoriedad usada (en otras palabras, P^* es un algoritmo determinístico si se fija s). Así, esta definición implica que el algoritmo de emulación puede extraer un testigo cuando el demostrador logra realizar una demostración exitosa. Esta propiedad implica la consistencia del esquema [37].

Esta definición puede formalizarse alternativamente mediante el siguiente experimento de indistinguibilidad de traza, Ind-tr, para un sistema de demostración $\Pi = (\text{SETUP}, \text{P}, \text{V})$:

Experimento 4 (Experimento de indistinguibilidad de traza)**.** $\text{Exp}^{\text{Ind-tr}}_{\mathscr{A},\Pi,E}(k)$*:*

1. *Se elige un bit al azar* $b \xleftarrow{\$} \{0,1\}$.
2. *Se ejecuta* $\omega \xleftarrow{\$} \text{SETUP}(k)$*, entregándose* ω *a* $\mathscr{A}$.
3. $\mathscr{A}$ *entrega una instancia* x *y un estado* s *para el demostrador* P *(que incluye la aleatoriedad a usar). Este demostrador puede desviarse del protocolo normal (dependiendo de* $\mathscr{A}$*).*
4. *Si* $b = 0$*, se ejecuta el protocolo de demostración entre* $\mathscr{P}$ *y* $\mathscr{V}$*. Se entrega la traza de este protocolo a* $\mathscr{A}$.
5. *Si* $b = 1$*, se ejecuta el emulador* E*, entregando la traza producida a* $\mathscr{A}$.
6. $\mathscr{A}$*, que emite un bit* b'.
7. *La salida del experimento se define como 1 si* $b = b'$*, y 0 si no.*

Así, todo sistema de demostración con emulación extendida con testigo posee la siguiente propiedad:

Definición 21 (Indistinguibilidad de traza)**.** *Un esquema de demostración* $(\text{SETUP}, \text{P}, \text{V})$ *posee* ***indistinguibilidad de traza*** *si para todo algoritmo determinístico de tiempo polinomial* P *existe un algoritmo de tiempo esperado polinomial* E*, que emula la interacción entre* P *y* V*, tal que para todo adversario de tiempo polinomial* $\mathscr{A}$*, existe una función despreciable* f *tal que:*

$$\mathbb{P}[\text{Exp}^{\text{Ind-tr}}_{\mathscr{A},\Pi,E}(k) = 1] \leq \frac{1}{2} + f(k)$$

Esta propiedad se obtiene directamente para un sistema que tenga de emulación extendida con testigo. Un adversario que logre que el experimento de indistinguibilidad de traza tenga salida 1 con probabilidad notablemente mayor a $1/2$ rompe la propiedad de emulación extendida con testigo. Finalmente, se define la ventaja Ind-tr de $\mathscr{A}$ como:

$$\text{Adv}^{\text{Ind-tr}}_{\Pi}(\mathscr{A}) = |2 \cdot \mathbb{P}[\text{Exp}^{\text{Ind-tr}}_{\mathscr{A},\Pi,E}(k) = 1] - 1|$$

La definición de extraibilidad permite expandir el significado de una demostración no interactiva: una demostración exitosa significa que el demostrador conoce cierta información secreta.

Definición 22 (Extraibilidad)**.** *Un sistema de demostración no interactivo* $(\text{SETUP}, \text{P}, \text{V})$ *es* ***extraíble*** *cuando existe un par de algoritmos* E_1, E_2 *tal que:*

- $(\omega_{ext}, \tau_e) \xleftarrow{\$} E_1(k)$

- $(\omega_{ext}, \tau_e) \xleftarrow{\$} E_1(k)$, *$\omega_{ext}$ es computacionalmente indistinguible de* $\omega_{crs} \xleftarrow{\$} \text{SETUP}(k)$.
- $\forall \mathscr{A}, (\omega_{ext}, \tau_e) \xleftarrow{\$} E_1(k), (x, \pi) \xleftarrow{\$} \mathscr{A}(\omega_{ext}, \tau_e)$, *la siguiente probabilidad es despreciable:*

$$\mathbb{P}[\text{V}(\omega_{crs}, x, \pi) = 1 \wedge (x, E_2(\omega_{ext}, \tau_e, x, \pi)) \notin R]$$

donde $\mathscr{A}$ es un adversario probabilístico de tiempo polinomial. En el caso en que esta probabilidad sea exactamente 0 y ω_{ext} siga la misma distribución de ω_{crs}, la propiedad se denomina ***extraibilidad perfecta***.

Indistinguibilidad de testigos y Nula Divulgación

Las definiciones de indistinguibilidad de testigos y de Nula Divulgación (Zero Knowledge) consideran la protección de la información secreta, poseída por el demostrador, del verificador.

Definición 23 (Indistinguibilidad de testigos)**.** *Un sistema de demostración* $(\text{SETUP}, \text{P}, \text{V})$ *posee* ***indistinguibilidad de testigos*** *si, para todo algoritmo* V *(potencialmente malicioso) y* (ω, x, w_1, w_2) *tales que:*

- $(x, w_1) \in R \wedge (x, w_2) \in R$
- $\omega \xleftarrow{\$} \text{SETUP}(k)$
- $tr_i \leftarrow \langle \text{P}(\omega, x, w_i), \text{V}(\omega, x) \rangle$, $i \in 1, 2$

se cumple que tr_1 es indistinguible de tr_2. Adicionalmente, V *puede recibir* (w_1, w_2)*. En el caso en que las distribuciones de tr_i sean idénticas, la propiedad se denomina* ***indistinguibilidad perfecta de testigos***.

Por otro lado, se dice que un sistema de demostración no interactivo $(\text{SETUP}, \text{P}, \text{V})$ *posee* ***indistinguibilidad de testigos*** *si, para todos* $(\omega_{crs}, x, w_1, w_2, \pi_1, \pi_2)$ *tales que:*

- $(x, w_1) \in R \wedge (x, w_2) \in R$
- $\omega_{crs} \xleftarrow{\$} \text{SETUP}(k)$
- $\pi_i \xleftarrow{\$} \text{P}(\omega_{crs}, x, w_i)$, $i \in 1, 2$

se cumple que (ω_{crs}, π_1) *es indistinguible de* (ω_{crs}, π_2)*. En el caso en que las distribuciones de las tuplas* (ω_{crs}, π_1) *y* (ω_{crs}, π_2) *sean idénticas, la propiedad se denomina* ***indistinguibilidad perfecta de testigos***.

En particular, la propiedad de Zero-Knowledge asegura que el verificador no obtiene información alguna sobre el testigo. De hecho, no obtiene ninguna información más allá del hecho de que lo que afirma el demostrador es cierto. Esto se logra vía demostrar que existe un algoritmo (denominado el "simulador"), el cual puede producir una interacción indistinguible de la producida por un demostrador honesto bajo ciertas condiciones (como poder rebobinar al verificador o tener acceso a parámetros secretos globales) que no debieran ocurrir en un escenario real.

En el caso de sistemas interactivos, el "verificador" puede desviarse de la ejecución normal de protocolo en cualquier manera que desee [34], ya que la propiedad sólo busca proteger al demostrador honesto. Se muestra además una versión particular de Zero Knowledge para sistemas interactivos: la propiedad de Zero Knowledge especial con verificador honesto. Esta definición es ligeramente más simple; sin embargo, es posible construir sistemas Zero Knowledge a partir de sistemas que sólo tienen la propiedad de Zero Knowledge especial con verificador honesto.

Definición 24 (Zero Knowledge especial con verificador honesto)**.** *Un sistema de demostración* $(\text{SETUP}, \text{P}, \text{V})$ *posee la propiedad de* ***Zero Knowledge especial con verificador honesto*** *si para todo adversario interactivo probabilístico de tiempo polinomial* $\mathscr{A}$ *existe un algoritmo simulador (no interactivo) S tal que:*

$$\begin{aligned}&\mathbb{P}[\omega \xleftarrow{\$} \text{SETUP}(k); (x,w,\rho) \xleftarrow{\$} \mathscr{A}(\omega); tr \leftarrow \langle \text{P}(\omega,x,w), \text{V}(\omega,x;\rho)\rangle : (x,w) \in R \wedge \mathscr{A}(tr) = 1]\\ \approx\ &\mathbb{P}[\omega \xleftarrow{\$} \text{SETUP}(k); (x,w,\rho) \xleftarrow{\$} \mathscr{A}(\omega); tr \leftarrow S(\omega,x,\rho) : (x,w) \in R \wedge \mathscr{A}(tr) = 1]\end{aligned}$$

En esta definición, ρ representa la aleatoriedad utilizada por V en su ejecución. Así, si se fija ρ, V se convierte en un algoritmo determinístico. Esta definición puede formalizarse mediante el siguiente experimento para un sistema de demostración $\Pi = (\text{SETUP}, \text{P}, \text{V})$:

Experimento 5 (Experimento de nula divulgación con verificador honesto)**.** $\text{Exp}^{\text{Ind-tr}}_{\mathscr{A},\Pi,S}(k)$*:*

1. *Se elige un bit al azar* $b \xleftarrow{\$} \{0,1\}$.
2. *Se ejecuta* $\omega_{crs} \xleftarrow{\$} \text{SETUP}(k)$*, entregándose* ω_{crs} *a* $\mathscr{A}$.
3. $\mathscr{A}$ *entrega una instancia x, un testigo w y una aleatoriedad* ρ *para el demostrador.*
4. *Si* $b = 0$*, se ejecuta el protocolo de demostración entre* $\mathscr{P}$ *y* $\mathscr{V}$*, entregando x a ambos, w a* P *y utilizando* ρ *como la aleatoriedad de* P*. Se entrega la traza a* $\mathscr{A}$.
5. *Si* $b = 1$*, se ejecuta el simulador S, entregándole* ω_{sim}*, x y* ρ*. Se entrega la traza obtenida a* $\mathscr{A}$*. En el caso de que* $(x,w) \notin R$ *se dice que la salida del experimento es automáticamente 1.*

6. *$\mathscr{A}$ emite un bit b'.*

7. *La salida del experimento se define como 1 si $b = b'$, y 0 si no.*

Así, una caracterización alternativa de la propiedad de Zero Knowledge con verificador honesto es la siguiente:

Definición 25 (Zero Knowledge especial con verificador honesto (caracterización alternativa))**.** *Un esquema de demostración* $(\text{SETUP}, \text{P}, \text{V})$ *posee la propiedad de* ***Zero Knowledge especial con verificador honesto*** *si para todo algoritmo determinístico de tiempo polinomial* P *existe un algoritmo de tiempo esperado polinomial E, que emula la interacción entre* P *y* V*, tal que para todo adversario de tiempo polinomial $\mathscr{A}$, existe una función despreciable f tal que:*

$$\mathbb{P}[\text{Exp}^{\text{Ind-tr}}_{\mathscr{A},\Pi,S}(k) = 1] \leq \frac{1}{2} + f(k)$$

Finalmente, se define la ventaja Ind-tr de $\mathscr{A}$ como:

$$\text{Adv}^{\text{Ind-tr}}_{\Pi}(\mathscr{A}) = |2 \cdot \mathbb{P}[\text{Exp}^{\text{Ind-tr}}_{\mathscr{A},\Pi,S}(k) = 1] - 1|$$

Finalmente, la propiedad de Zero Knowledge de sistemas de demostración se define como sigue:

Definición 26 (Zero Knowledge)**.** *Un sistema de demostración* $(\text{SETUP}, \text{P}, \text{V})$ *posee la propiedad de* ***Zero Knowledge*** *si para todo algoritmo interactivo probabilístico de tiempo polinomial V existe un algoritmo simulador (no interactivo) $M^{V(\cdot)}$ tal que las distribuciones $tr \leftarrow M^{V(x)}$ y $tr \leftarrow \langle \text{P}(\omega, x, w), \text{V}(\omega, x) \rangle$ son computacionalmente indistinguibles. Se dice que M es un algoritmo simulador (no interactivo) que toma a* V *como oráculo. En el caso en que estas distribuciones sean idénticas para todo (x, w), esta propiedad se denomina* ***Zero-Knowledge perfecta****.*

Por otro lado, un sistema de demostración no interactivo $(\text{SETUP}, \text{P}, \text{V})$ *posee la propiedad de* ***Zero Knowledge*** *si existe un par de algoritmos S_1, S_2 tal que:*

- $(\omega_{sim}, \tau_s) \xleftarrow{\$} S_1(k)$ *: S_1 es un algoritmo que, dado un parámetro de seguridad k como entrada, retorna una cadena ω_{sim} y una puerta secreta τ_s. El valor ω_{sim} es una cadena de referencia común* simulada*.*

- $\pi_s \xleftarrow{\$} S_2(\omega_{sim}, \tau_s, x)$ *: S_2 es un algoritmo que, dada la cadena de referencia común simulada, la puerta secreta y una instancia x, genera una* demostración simulada π_s *(sin necesidad de un testigo).*

- *Las siguientes interacciones son indistinguibles para todo adversario probabilístico de tiempo polinomial $\mathscr{A}$:*

  - $\omega_{crs} \xleftarrow{\$} \text{SETUP}(k)$*; luego, se entrega ω_{crs} y acceso a un oráculo* $\text{P}(\omega_{crs}, \cdot, \cdot)$ *a $\mathscr{A}$.*

– $(\omega_{sim}, \tau_s) \xleftarrow{\$} S_1$*; luego, se entrega* ω_{sim} *y acceso a un oráculo* $S(\omega_{sim}, \tau_s, \cdot, \cdot)$ *a* $\mathscr{A}$*. El algoritmo S, al recibir una entrada* (x,w)*, retorna* $S_2(\omega_{sim}, \tau_s, x)$

donde $\mathscr{A}$ *es un adversario probabilístico de tiempo polinomial. Se considera que ambos oráculos,* P *y S retornan* $\perp$ *si reciben una entrada* $(x,w) \notin R$*. En el caso en que las distribuciones de estas interacciones son idénticas para todo* (x,w)*, esta propiedad se denomina* ***Zero-Knowledge perfecta****.*

Adicionalmente, en el caso de sistemas no interactivos, puede construirse una variante más fuerte de la propiedad de Zero Knowledge, para permitir su uso en conjunto (por ejemplo, de manera concurrente) con otros sistemas [41]:

Definición 27 (Zero Knowledge componible)**.** *Un sistema de demostración no interactivo* $(\text{SETUP}, \text{P}, \text{V})$ *posee la propiedad de* ***Zero Knowledge componible*** *si existe un par de algoritmos* S_1*,* S_2 *tal que:*

- $(\omega_{sim}, \tau_s) \xleftarrow{\$} S_1(k)$*: similar al caso anterior,* S_1 *genera una cadena de referencia común y una*
- $\pi_s \xleftarrow{\$} S_2(\omega_{sim}, \tau_s, x)$
- $\forall \mathscr{A}$*, las siguientes dos probabilidades son idénticas:*

$$\mathbb{P}[(\omega,\tau) \xleftarrow{\$} \text{S}_1(k); (x,w,s) \xleftarrow{\$} \mathscr{A}(\omega,\tau); \pi \xleftarrow{\$} \text{P}(\omega,x,w) : \mathscr{A}(\pi,s) = 1]$$
$$\mathbb{P}[(\omega,\tau) \xleftarrow{\$} \text{S}_1(k); (x,w,s) \xleftarrow{\$} \mathscr{A}(\omega,\tau); \pi \xleftarrow{\$} \text{S}_2(\omega,x,\tau) : \mathscr{A}(\pi,s) = 1]$$

Se considera que P *y* S_2 *retornan* $\perp$ *si* $\mathscr{A}$ *emite un par* $(x,w) \notin R$*.*

Las combinaciones de estas diferentes propiedades dan origen a diversas denominaciones para los sistemas de demostración existentes:

Definición 28 (Sistemas de demostración)**.** *Un trío de algoritmos* $(\text{SETUP}, \text{P}, \text{V})$ *es un* ***sistema de demostración*** *para una relación R asociada a un conjunto L si posee completitud y consistencia. Adicionalmente:*

- *Si* P *y* V *son protocolos no interactivos, se dice que es* ***no interactivo*** *(NI).*
- *Si posee la propiedad de extraibilidad, se dice que es* ***extraíble****. En algunos trabajos sobre sistemas interactivos se denomina a esta propiedad como* consistencia especial*.*
- *Si posee la propiedad de indistinguibilidad de testigos, se dice que es de* ***testigos indistinguibles*** *(witness-indistinguishable, WI; NIWI en el caso de sistemas no interactivos).*

- *Si posee la propiedad de Zero Knowledge, se dice que es **Zero Knowledge** (ZK, NIZK en el caso de sistemas no interactivos).*
- *Si posee la propiedad de indistinguibilidad de testigos y extraibilidad, se dice que es un **sistema de demostración de conocimiento de testigos indistinguibles** (WIPoK, NIWIPoK en el caso de sistemas no interactivos).*
- *Si posee la propiedad de Zero Knowledge y extraibilidad, se dice que es un **sistema de demostración de conocimiento Zero Knowledge** (ZKPoK, NIZKPoK en el caso de sistemas no interactivos).*

1.3.8 Redes de mezcla

Las redes de mezcla constituyen una categoría particular de redes de comunicación anónima. Una red de mezcla es una primitiva criptográfica de clave pública, que toma como entrada un número de textos cifrados, y entrega una permutación aleatoria de los textos planos correspondientes [24].

La implementación usual de una red de mezcla consiste en múltiples entidades (los *servidores de mezcla*), de modo que la información sobre la permutación aplicada a los textos cifrados se encuentra repartidos entre éstas. Así, la correspondencia entre textos planos y textos cifrados se mantiene oculta incluso si uno o más de los servidores de mezcla han sido comprometidos por un adversario.

Se consideran las siguientes entidades participantes de una red de mezcla:

Usuarios: las entidades que emiten los mensajes a ser mezclados por la red.

Servidores de desencriptación: las entidades que reciben los textos cifrados aleatoriamente permutados y realizan la desencriptación distribuida de éstos.

Servidores de mezcla: las entidades que construyen la permutación aplicada al vector de textos cifrados ingresado a la red.

Adicionalmente, se considera la existencia de al menos tres protocolos para una red de mezcla: KEYGEN, el protocolo de generación de claves; ENCRYPT, el protocolo para la encriptación de mensajes; y EXEC, el protocolo distribuido consistente en la operación de mezclado y desencriptación de la red de mezcla.

Existen múltiples definiciones de seguridad para una red de mezcla y sus diferentes propiedades; se enuncian a continuación dos modelos relevantes para la discusión: el modelo basado en simulaciones y el modelo de composición universal.

Modelo basado en Simulaciones

En este modelo pueden definirse diversas propiedades para una red de mezcla; sin embargo, este trabajo se enfocará en la definición de privacidad. En particular, la definición de seguridad presentada en [24] se define en base al siguiente experimento:

Experimento 6 (Experimento de anonimato para redes de mezcla)**.** $\text{ANON}_{\mathscr{A},\mathscr{MIX}}(k)$*:*

1. *La red de mezcla $\mathscr{MIX}$ genera sus claves públicas y secretas utilizando $\mathscr{MIX}$.*KEYGEN*, anunciándose las claves públicas.*
2. *El adversario $\mathscr{A}$ computa y emite dos listas de mensajes en representación de los usuarios honestos: $M_0 = \{m_1^0, ..., m_{k_h}^0\}$ y $M_1 = \{m_1^1, ..., m_{k_h}^1\}$, donde una es una permutación de la otra.*
3. *Se escoge un bit $b \xleftarrow{\$} \{0,1\}$. Se entregan los mensajes de la lista M_b a los correspondientes usuarios honestos. Los usuarios encriptan sus mensajes utilizando $\mathscr{MIX}$.*ENCRYPT*, publicando los textos cifrados c_i. El adversario emite textos cifrados arbitrarios en representación de los usuarios corruptos.*
4. *La red de mezcla procesa los textos cifrados publicados utilizando $\mathscr{MIX}$.*EXEC*, y emite una lista (mezclada) de mensajes $M' = \{m'_1, ..., m'_k\}$.*
5. *El adversario intenta distinguir cuál de las listas se eligió y emite un bit $b' \in \{0,1\}$.*
6. *La salida del experimento se define como 1 si $b = b'$, y 0 si no.*

Con esto, se define la ventaja ANON del adversario $\mathscr{A}$ como:

$$\text{Adv}^{\text{ANON}}_{\mathscr{MIX}}(\mathscr{A}) = |2 \cdot \mathbb{P}[\text{ANON}_{\mathscr{A},\mathscr{MIX}}(k) = 1] - 1|$$

Finalmente, una red de mezcla posee *anonimato* si cumple la siguiente definición:

Definición 29 (Anonimato de una red de mezcla en el modelo basado en simulaciones)**.** *Una red de mezcla posee anonimato ante un conjunto de adversarios $\mathbb{U}$ si para cualquier adversario $\mathscr{A} \in \mathbb{U}$, existe una función despreciable f tal que:*

$$\text{Adv}^{\text{ANON}}_{\mathscr{MIX}}(\mathscr{A}) \leq f(k)$$

Modelo de Composición Universal

Por otro lado, en el Modelo de Composición Universal [19], se tienen dos modelos de computación. Por un lado, un modelo de Mundo Real (que corresponde al escenario real, con un protocolo criptográfico entre múltiples entidades) y el modelo de Mundo Ideal. Ambos modelos poseen las mismas entidades participantes; sin embargo, mientras que en el Mundo Real los cálculos se realizan a través de los protocolos del sistema criptográfico, en el Mundo Ideal todas las entidades entregan su información a una entidad confiable llamada la *Funcionalidad Ideal*, que realiza los cálculos deseados.

Por otro lado, existe una entidad especial Z, llamada el *Ambiente*, que se encuentra fuera de los modelos, escoge los datos de entrada de las demás entidades (por ejemplo, los mensajes que los usuarios se envían) y recibe los resultados de los protocolos (recibidos por los usuarios).

Se dice que un protocolo Π *concreta una funcionalidad* $\mathscr{F}$ *de manera segura* (o que Π es *UC-seguro*) si no existe un ambiente que pueda distinguir si está interactuando con las entidades del Mundo Real (que ejecutan Π) o con las entidades del Mundo Ideal a través de la funcionalidad ideal $\mathscr{F}$. Esto garantiza diversas propiedades; por ejemplo, que múltiples instancias del protocolo pueden ejecutarse de forma concurrente sin ninguna pérdida de seguridad.

Modelo de comunicación

Las condiciones de comunicación asumidas para una red de mezcla en este trabajo son las siguientes:

Entrega garantizada de mensajes: Se asume que la comunicación es asíncrona, autentificada y que se garantiza la entrega de los mensajes enviados (por ejemplo, los mensajes no pueden ser descartados por un adversario que interviene el canal de comunicación).

Existencia de un Diario Mural o Bulletin Board: Se utiliza un diario mural autentificado, en el que los mensajes no pueden ser borrados, como mecanismo de comunicación entre las partes. Idealmente, el diario mural transmite toda la información que recibe, junto con las identidades de los emisores, a todas las entidades presentes.

Seguridad UC de una red de mezcla

En las condiciones de comunicación ya definidas, puede construirse la siguiente funcionalidad ideal $\mathscr{F}_{MIX}$ [18]:

1. Al inicializarse, se espera a que m valores de la forma $(\texttt{Mix-Server}, id)$ se reciban de $M_1 ... M_m$ y n valores $(\texttt{Dec-Server}, id, D_i)$ de $D_1 ... D_n$. Cuando se recibe un mensaje $(\texttt{Dec-Server}, id, D_i)$, se emite a todas las demás entidades. Se construye una lista D de los servidores de desencriptación.

2. Se escoge una ID de sesión sid; se envía el mensaje $(\texttt{Open}, id, sid)$ a todas las entidades.

3. Se construyen listas vacías K y U. Se espera a que k valores se reciban desde los usuarios $U_1 ... U_k$, con la forma $(\texttt{Send}, id, sid, m_i)$. Al recibir cada valor, se agrega un mensaje m_i a la lista K, un ID de usuario U_i a la lista U, y se envía $(\texttt{Send}, id, sid, U_i)$ a todas las entidades.

4. Se escoge una permutación aleatoria π. Se aplica π a la lista K, obteniéndose una lista K'. Se entrega K' al adversario. El adversario puede responder con un mensaje $(\texttt{ok})$ o con $(\texttt{re-shuffle}, D_i, U^*)$, donde D_i corresponde a un servidor corrupto en D, y U^* corresponde a un subconjunto de usuarios corruptos de U.

5. Si el adversario respondió `re-shuffle`, se elimina D_i de D, se eliminan todos los usuarios en U^* de U, y se eliminan los mensajes enviados por estos usuarios de K. Se repite el paso 4.

6. Si el adversario respondió `ok`, se envía $(\texttt{Output}, id, sid, K', U)$ a todas las entidades.

7. Se repite el proceso, comenzando desde el paso 2.

Con esto, queda definida la seguridad de una red de mezcla:

Definición 30 (Seguridad de una red de mezcla)**.** *Un protocolo de red de mezcla $\mathscr{MIX}$ es UC-seguro para un conjunto de adversarios $\mathbb{A}$ si para todo adversario del Mundo Real $\mathscr{A} \in \mathbb{A}$, existe un adversario del Mundo Ideal $\mathscr{B}$ tal que no existe un ambiente que pueda distinguir si está interactuando con $\mathscr{A}$ y la implementación del Mundo Real de $\mathscr{MIX}$ o si está interactuando con $\mathscr{B}$ e $\mathscr{F}_{MIX}$, con una probabilidad no despreciable.*

1.4 Organización de esta Memoria

Se presenta a continuación la organización de los demás capítulos de esta memoria.

El Capítulo 2 presenta de manera concreta y criptográficamente completa el problema que se tratará y el tipo de soluciones que se considerarán para este trabajo, explicando las interacciones existentes en la solución a construir, así como el estándar de seguridad que se plantea alcanzar.

El Capítulo 3 contiene gran parte del trabajo realizado a lo largo de esta investigación. Se analizan distintas alternativas de solución, siendo necesario realizar un análisis de distintos tipos de herramientas relacionadas con la problemática a resolver. Se estudiaron diversas variantes de

dinero electrónico, además de esquemas de firmas, como componentes de una posible solución. Se estudiaron ventajas y desventajas presentes al intentar utilizar estos esquemas como base para una solución, proceso que guió la búsqueda de nuevas herramientas.

El Capítulo 4 consiste en la descripción de la solución construida, basada en algunas de las herramientas estudiadas anteriormente.

El Capítulo 5 detalla las demostraciones de seguridad que validan la solución propuesta en la sección anterior, validando de esta forma la correspondencia entre el problema a resolver y esta solución. Se dejan ciertas demostraciones para el Apéndice de este documento (en particular, aquellas que son utilizadas como herramientas para las demostraciones principales).

El Capítulo 6 detalla las conclusiones del trabajo, planteándose posibles expansiones y mejoras para el esquema presentado.

Capítulo 2

Especificación del Problema

2.1 Motivación

Se desea que el sistema de comunicación pueda permitir el filtrado de mensajes según la política (públicamente conocida) de cada una de las entidades que realizan la transmisión de éstos. Por ejemplo, un servidor de mezcla puede requerir que ciertas autoridades específicas certifiquen los mensajes que pasan por él, por razones legales, financieras o administrativas. Se requiere que este filtrado sea seguro: es decir, que ninguno de los mensajes que llegue al final de la red viole alguna de las políticas de los servidores por los cuales pasó.

Por otro lado, todos los participantes del sistema debieran poder asegurarse de que este filtrado se realiza de forma correcta: un usuario que, por ejemplo, pague por el acceso al sistema de comunicación deseará asegurarse de obtener garantías de uso.

Finalmente, es necesario preservar las propiedades de un sistema usual de comunicación anónima. Los usuarios deben obtener confidencialidad, incluso con respecto a las autoridades certificadoras. Por ejemplo, sistemas gubernamentales de denuncia necesitarían que el anonimato se mantenga incluso ante las autoridades de gobierno, de modo de poseer un grado de credibilidad.

La formalización del concepto de una *red de mezcla con verificación de mensajes certificados* busca aunar estos criterios de la forma que se presenta a continuación.

2.2 Formalización

Se busca construir una red de comunicación anónima basada en reencriptación que permita la verificación de una propiedad criptográfica en los textos cifrados circulantes en ésta. Se denomina a la siguiente construcción una red de mezcla con verificación de mensajes certificados.

En el protocolo participan t usuarios $U_1, ..., U_t$, m servidores de mezcla $M_1, ...M_m$ y n servidores de desencriptación $D_1, ..., D_n$. Adicionalmente, una autoridad verificadora $\mathscr{K}$ realiza la verificación de las operaciones y denuncia a los participantes deshonestos. Finalmente, se tiene un conjunto de s autoridades $S_1, ..., S_s$ que realizan el firmado de textos cifrados.

Se define que esta red debe poseer los siguientes protocolos que involucran a las múltiples entidades presentes:

KEYGEN: Como su nombre lo indica, este protocolo genera las claves de encriptación y desencriptación del sistema, así como las claves de firma para las entidades certificadoras de mensajes.

ENCRYPT: Este protocolo permite la encriptación de los mensajes a ser ingresados a la red. Se denomina la lista de mensajes correctamente ingresados a la red como la *entrada* de ésta.

EXEC: A través de este protocolo, ejecutado por todos los servidores, se realiza el mezclado y la desencriptación (de forma distribuida) de los textos cifrados; finalmente, se emite una lista de textos planos K', que contiene una permutación aleatoria de los mensajes ingresados a la red.

Además, con el objetivo de definir la propiedad criptográfica a exigir en los textos cifrados presentes en la red, se establece la presencia de los siguientes algoritmos criptográficos:

SIGN: Este algoritmo permite que una autoridad genere una firma sobre un texto cifrado, de forma de que pueda ingresar a la red de mezcla. Este algoritmo es utilizado en el protocolo ENCRYPT.

VERIFY: Este algoritmo permite que los servidores de mezcla verifiquen la validez de los textos cifrados que reciben (en particular, la validez de la firma adjunta a éstos). Este algoritmo es utilizado en el protocolo EXEC.

Se requiere, además, que la validez de estas firmas puedan ser chequeadas públicamente (es decir, por cualquier entidad participante en el sistema) en cada paso del proceso de mezclado, sin

comprometer la seguridad del sistema. Finalmente, se requiere que los textos cifrados (y las firmas adjuntas) constituyan un esquema de encriptación realeatorizable, de modo de ser compatible con el proceso de mezclado.

2.3 Propiedades criptográficas

Para definir la seguridad de una red de mezcla con verificación de mensajes certificados se utilizará lo expuesto sobre anonimato de redes de mezcla expuesto en el Capítulo 1, incorporando además conceptos de filtrado.

Intuitivamente, incluso si un adversario pudiera corromper a un subconjunto de las autoridades certificadoras, a un subconjunto de los usuarios y a un subconjunto (estricto) de los servidores de mezcla, no puede determinar quiénes han enviado ciertos mensajes; tampoco puede lograr que un mensaje que no ha sido apropiadamente certificado sea procesado por la red de mezcla. Considerando esto, se definirá el significado de la verificación de mensajes certificados en el contexto de una red de mezcla, via el siguiente experimento:

Experimento 7 (Experimento de anonimato e infalsificabilidad para redes de mezcla certificadas)**.** ANON-UF$_{\mathscr{A},\mathscr{MIX}}(k)$*:*

1. *La red de mezcla $\mathscr{MIX}$ genera sus claves públicas y secretas utilizando el procedimiento $\mathscr{MIX}$.*KeyGen*, anunciándose las claves públicas.*
2. *El adversario $\mathscr{A}$ emite un subconjunto P_i de las claves de verificación publicadas por $\mathscr{MIX}$.*KeyGen *para cada servidor de mezcla M_i. Se denota $P \leftarrow \bigcup P_i$. El servidor de mezcla i sólo mezcla en su salida los mensajes que posean firmas para cada cada clave en P_i.*
3. *El adversario $\mathscr{A}$ computa y emite dos listas de mensajes en representación de los usuarios honestos: $M_0 = \{m_1^0, ..., m_{k_h}^0\}$ y $M_1 = \{m_1^1, ..., m_{k_h}^1\}$, donde una es una permutación de la otra.*
4. *Se escoge un bit $b \xleftarrow{\$} \{0,1\}$. Se entregan los mensajes de la lista M_b a los correspondientes usuarios honestos. Los usuarios encriptan sus mensajes utilizando $\mathscr{MIX}$.*Encrypt*, obteniendo firmas sobre éstos a través de $\mathscr{MIX}$.*Sign*. De esta forma, se publican los textos cifrados y firmas (c_i, σ_i).*
5. *El adversario emite textos cifrados y firmas arbitrarias en representación de los usuarios corruptos, teniendo acceso a un oráculo $\mathscr{MIX}$.*Sign*. Este oráculo recibe un texto cifrado c y una de las claves de verificación de firma vk construidas por $\mathscr{MIX}$.*KeyGen*, agregando el par m a una lista SM. Retorna una firma sobre c válida bajo la clave de verificación vk.*

6. *La red de mezcla procesa los textos cifrados publicados utilizando $\mathscr{M}\mathscr{I}\mathscr{X}$.*EXEC*, y emite una lista (mezclada) de mensajes $M' = \{m'_1, ..., m'_k\}$.*

7. *El adversario intenta distinguir cuál de las listas se eligió y emite un bit $b' \in \{0,1\}$.*

8. *La salida del experimento es 1 si $b = b'$, y 0 de lo contrario.*

Este modelo requiere que todos los usuarios honestos firmen sus mensajes de la misma forma (es decir, que obtengan el total de las firmas necesarias según las políticas de todos los servidores de mezcla a utilizar). Esto, porque un adversario malicioso podría violar el anonimato de mensajes que carguen firmas para diferentes conjuntos de entidades, ya que éstas son públicamente verificables. La seguridad del sistema ante el adversario $\mathscr{A}$ se define en dos partes. Por un lado, la salida del experimento determina la ventaja MIX-ANON del adversario:

$$\mathrm{Adv}^{\text{MIX-ANON}}_{\mathscr{M}\mathscr{I}\mathscr{X}}(\mathscr{A}) = |2 \cdot \mathbb{P}[\text{ANON-UF}_{\mathscr{A},\mathscr{M}\mathscr{I}\mathscr{X}}(k) = 1] - 1|$$

Por otro lado, la ventaja MIX-UF del adversario, $\mathrm{Adv}^{\text{MIX-UF}}_{\mathscr{M}\mathscr{I}\mathscr{X}}(\mathscr{A})$, se define como la siguiente probabilidad, dado el mismo experimento:

$$\mathrm{Adv}^{\text{MIX-UF}}_{\mathscr{M}\mathscr{I}\mathscr{X}}(\mathscr{A}) = \mathbb{P}[P \neq \emptyset \wedge \exists m^* \in M' \;\; m^* \notin SM]$$

De esta forma, queda definido el anonimato e infalsificabilidad de una red de mezcla certificada con verificación de mensajes certificados:

Definición 31 (Anonimato e infalsificabilidad de una red de mezcla certificada)**.** *Una red de mezcla con verificación de mensajes certificados posee anonimato ante un conjunto de adversarios $\mathbb{U}$ si para cualquier adversario $\mathscr{A}$ de este conjunto, existe una función despreciable f tal que:*

$$\mathrm{Adv}^{\text{MIX-ANON}}_{\mathscr{M}\mathscr{I}\mathscr{X}}(\mathscr{A}) \leq f(k)$$

Análogamente, una red de mezcla con verificación de mensajes certificados posee infalsificabilidad ante un conjunto de adversarios $\mathbb{U}$ si para cualquier adversario $\mathscr{A}$ de este conjunto, existe una función despreciable g tal que:

$$\mathrm{Adv}^{\text{MIX-UF}}_{\mathscr{M}\mathscr{I}\mathscr{X}}(\mathscr{A}) \leq g(k)$$

Finalmente, una red de mezcla con verificación de mensajes certificados es segura ante una clase de adversarios si posee ambas propiedades, anonimato e infalsificabilidad ante este conjunto de adversarios.

Esta definición de seguridad es la que se buscará en un eventual esquema a construir.

Capítulo 3

Exploración de Soluciones

3.1 Condiciones para una posible solución

Conjugar en un mismo esquema firmas y encriptación no es una tarea trivial. Existen diferentes formas de combinar estos esquemas [1], con diferentes características:

- Firmar un texto plano y adjuntar la firma generada al texto cifrado viola la privacidad del texto cifrado, ya que enlaza al texto cifrado con un texto plano en particular (aquel con el cual la firma verifica correctamente).
- Firmar un texto plano y luego encriptar ambos componentes impide la verificación de la firma (ya que el algoritmo de verificación requiere el mensaje firmado) antes de la desencriptación. Adicionalmente, implementaciones ingenuas de esta modalidad permiten que un adversario realice el reenvío de un mensaje utilizando la misma firma adjunta, pero encriptando el texto con respecto a otra clave: en otras palabras, un mensaje de una entidad $\mathscr{A}$ hacia $\mathscr{B}$ puede ser desencriptado por $\mathscr{B}$ y encriptado con la llave pública de $\mathscr{C}$. Esto puede interpretarse como la generación de un mensaje de $\mathscr{A}$ para $\mathscr{C}$ sin su autorización. Este reenvío no es detectable, lo que puede suponer problemas para la aplicación.
- Encriptar el texto plano y luego generar una firma sobre éste evita los problemas anteriores. Sin embargo, un adversario pudiera reemplazar una firma adjunta a un mensaje por una de su elección. Por ejemplo, si los mensajes son firmados para demostrar la autoría de un mensaje, un adversario pudiera cambiar una firma sobre un texto cifrado por una propia, pudiendo afirmar (falsamente) que es el autor real del mensaje.

Adicionalmente, existen diversos esfuerzos por combinar estos mecanismos de forma más natural y/o eficiente [47] [50] [29]. Esta problemática debe ser tratada con cuidado al momento de

construir una solución al problema planteado.

Por otro lado, debe asegurarse que un usuario honesto (es decir, cuyos mensajes no debieran ser filtrados por la red) siempre logran ingresar sus mensajes exitosamente a ésta, no siendo retenidos por un servidor de mezcla malicioso. En caso contrario, un servidor podría aliarse con un adversario y filtrar mensajes de modo de comprometer el anonimato de estos usuarios.

3.2 Firmas encriptadas verificables

El concepto de firmas encriptadas verificables, definido según lo expuesto en [15] extiende el concepto de un esquema de firmas, permitiendo su encriptación y verificación, de modo que puede chequearse la presencia de una firma sobre un mensaje sin conocer la firma en sí. De esta forma, esto puede constituir una propiedad encriptada y verificable que podría tenerse sobre los mensajes circulantes en una red de comunicación anónima.

Definición 32 (Firmas encriptadas verificables)**.** *Un esquema de firmas encriptadas verificables es una tupla de algoritmos* (KEYGEN, SIGN, VERIFY, ADJKEYGEN, VESIGN, VESIGNVERIFY, ADJUDICATE)*, donde* (KEYGEN, SIGN, VERIFY) *son un esquema de firmas, y cumple:*

> ADJKEYGEN(1^k)*: genera un par de claves, pública y privada,* (avk, ask) *para una nueva entidad, el* adjudicador*.*
>
> VESIGN$_{sk,avk}(M)$*: genera, probabilísticamente, una firma encriptada verificable* ω *sobre* M*.*
>
> VESIGNVERIFY$_{vk,avk}(M,\omega)$*: De manera análoga a* VERIFY*, determina si una firma encriptada verificable sobre* M *es válida para las claves dadas.*
>
> ADJUDICATE$_{avk,ask,vk}(\omega)$*: Emite una firma (común y corriente)* σ *sobre* M*, válida bajo* vk*.*

Más formalmente, la condición de correctitud para un esquema de firmas encriptadas verificables requiere:

$$\begin{aligned} \text{VESIGNVERIFY}_{vk,avk}(M, \text{VESIGN}_{sk,apk}(M)) &= 1 \\ \text{VERIFY}_{vk}(M, \text{ADJUDICATE}_{avk,ask,vk}(\text{VESIGN}_{sk,avk}(M))) &= 1 \end{aligned}$$

Este mecanismo criptográfico permite a una entidad mostrar a otra que posee una firma, sin revelarla. Sin embargo, las firmas encriptadas verificables siempre se crean utilizando el mensaje, por lo que la autoridad certificadora podría ver el mensaje que está firmando. No es claro cómo remediar esto: una forma ingenua de hacerlo es remover la dependencia entre la firma y el mensaje; sin embargo, esto permite la creación de firmas falsas a un adversario.

Adicionalmente, la posibilidad de reutilización de firmas por parte de adversarios indica que es necesario el estudio de esquemas que limiten el uso de ciertos *tokens* de autorización. Los esquemas de dinero electrónico, o *E-Cash*, son una interesante alternativa para analizar mejor las problemáticas a resolver.

3.3 Esquemas de E-Cash

El dinero electrónico o *E-Cash* con las propiedades que utilizan hoy en día, fue concebido por Chaum en 1982 [22]. Consiste en un sistema con tres tipos de entidades: usuarios $\mathscr{U}$, mercantes $\mathscr{M}$ y un banco $\mathscr{B}$. Un usuario puede retirar monedas de su cuenta bancaria; posteriormente, utilizando estas monedas, puede entregar a un mercante un pago por, por ejemplo, la prestación de un servicio, de modo que el pago puede ser depositado en la cuenta bancaria del mercante. Se llama *moneda* tanto a la información utilizada por $\mathscr{U}$ para hacer las transferencias, como a la información recibida por $\mathscr{M}$ al recibir un pago y al realizar el depósito.

La definición formal de los esquemas de E-Cash varía según las técnicas utilizadas para proveerles de seguridad; además, diferentes variantes agregan algoritmos y protocolos adicionales con el objetivo de agregar diferentes funcionalidades y/o garantías al sistema. Sin embargo, las garantías básicas de seguridad tienden a ser las mismas:

Infalsificabilidad: de manera similar a firmas digitales, se requiere que sólo $\mathscr{B}$ pueda crear monedas válidas, y que sólo un usuario $\mathscr{U}$ válido pueda retirar y depositar monedas desde y hacia su cuenta bancaria.

Desvinculación: se requiere que sea difícil para $\mathscr{B}$ distinguir si dos pagos fueron realizados por el mismo usuario $\mathscr{U}$ (a menos que estos pagos gasten más que el saldo disponible).

Anonimato de pago: se requiere que $\mathscr{B}$ no pueda rastrear las monedas. Puede formalizarse de diferentes formas, en general, apuntando a que no pueda deducirse la historia de una moneda.

Protección ante doble gasto: se requiere que una moneda no pueda ser gastada más de una vez.

3.3.1 E-Cash compacto

Un importante avance en la factibilidad de uso de un sistema de E-Cash fue el esquema de E-Cash compacto propuesto por Camenisch, Hohenberg y Lysyanskaya [16]. Este esquema permite que un usuario tenga en su poder una cantidad k de monedas utilizando espacio $O(log(k))$. La definición

formal de E-Cash propuesta por estos autores se acerca bastante a lo expuesto de forma informal en la sección anterior. En este esquema, los mercantes son un subconjunto de los usuarios.

Dado que esta definición de E-Cash involucra protocolos, se denotará $\mathrm{P}(\mathscr{A}(x),\mathscr{B}(y))$ como el protocolo P entre $\mathscr{A}$ y $\mathscr{B}$, que reciben entradas x e y, respectivamente.

La siguiente es la definición informal de un esquema de E-Cash compacto.

Definición 33 (Esquema de E-Cash compacto)**.** *Una tupla de algoritmos* (BKEYGEN, UKEYGEN, WITHDRAW, SPEND, DEPOSIT, IDENTIFY, VERIFYGUILT, TRACE, VERIFYOWNERSHIP) *es un esquema de E-Cash Compacto si:*

- $\mathrm{BKEYGEN}(1^k)$ *genera un par de claves* $(pk_{\mathscr{B}},sk_{\mathscr{B}})$ *para* $\mathscr{B}$.
- $\mathrm{UKEYGEN}(1^k)$ *genera un par de claves* $(pk_{\mathscr{U}},sk_{\mathscr{U}})$ *para* $\mathscr{U}$. *Los mercantes también utilizan este algoritmo para obtener sus claves.*
- $\mathrm{WITHDRAW}(\mathscr{U}(pk_B,sk_U,n),\mathscr{B}(pk_U,sk_B,n))$ *permite que un usuario* $\mathscr{U}$ *obtenga una billetera* W *de* n *monedas, interactuando con el banco* $\mathscr{B}$. *La salida del protocolo es, para* $\mathscr{U}$, *la billetera* W *o un mensaje de error; para* $\mathscr{B}$, *información* T_W *que permitirá identificar al usuario en caso de doble gasto. El banco mantiene una base de datos* D *para esta información, en la que almacena* $(pk_{\mathscr{U}},T_W)$.
- $\mathrm{SPEND}(\mathscr{U}(W,pk_{\mathscr{M}}),\mathscr{M}(sk_{\mathscr{M}},pk_{\mathscr{B}},n))$ *permite que* $\mathscr{U}$ *transfiera una de las monedas de su billetera a* $\mathscr{M}$. $\mathscr{M}$ *obtiene un número serial* S *de la moneda, y una demostración* π *de la validez de ésta.* $\mathscr{U}$ *obtiene una billetera actualizada* W'.
- $\mathrm{DEPOSIT}(\mathscr{M}(sk_{\mathscr{M}},S,\pi,pk_{\mathscr{B}}),\mathscr{B}(pk_{\mathscr{M}},sk_{\mathscr{B}}))$ *permite que* $\mathscr{M}$ *deposite una moneda* (S,π) *en su cuenta en el banco* $\mathscr{B}$. *En el caso en que* (S,π) *haya sido obtenida a través de* SPEND, *la moneda será aceptada por el banco. El banco actualiza la lista* L *de monedas gastadas, agregando* (S,π). $\mathscr{M}$ *obtiene un mensaje de error, o nada.*
- $\mathrm{IDENTIFY}(S,\pi_1,\pi_2)$ *permite identificar a aquellos usuarios que hayan gastado una moneda dos veces, usando un número serial* S *y dos demostraciones de validez* π_1,π_2, *posiblemente entregadas por mercantes maliciosos. La salida de este algoritmo es una clave pública* $pk_{\mathscr{U}}$ *y* Π_G. *Si los mercantes que entregaron* π_1 *y* π_2 *no eran maliciosos,* Π_G *es evidencia de que* $pk_{\mathscr{U}}$ *es la clave pública de un usuario que gastó una moneda* S *dos veces.*
- $\mathrm{VERIFYGUILT}(S,pk_{\mathscr{U}},\Pi_G)$ *permite verificar públicamente que el usuario con clave pública* $pk_{\mathscr{U}}$ *es culpable de gastar* S *dos veces.*
- $\mathrm{TRACE}(S,pk_{\mathscr{U}},\Pi_G,D,n)$ *permite obtener los números seriales* $S_1...S_m$ *de todas las monedas entregadas a* $\mathscr{U}$, *con demostraciones* $\Pi_1...\Pi_M$ *de pertenencia de éstas a* $\mathscr{U}$. *Si* $\mathscr{U}$ *era honesto, este algoritmo no hace nada.*
- $\mathrm{VERIFYOWNERSHIP}(S,\Pi,pk_{\mathscr{U}},n)$ *permite verificar la demostración* Π *de que la moneda* S *pertenece a un usuario con clave pública* $pk_{\mathscr{U}}$, *que realizó un doble gasto.*

Sobre esta definición de un esquema de E-Cash, se pueden definir las siguientes condiciones de seguridad:

Correctitud: al ejecutarse WITHDRAW entre usuarios honestos y un banco honesto, ninguno debiera obtener un mensaje de error. Análogamente, no debieran generarse errores al ejecutarse SPEND entre usuarios y mercantes honestos, siendo la moneda aceptada por el mercante.

Balance: ningún conjunto de usuarios y mercantes debiera ser capaz de gastar más monedas que el total obtenido del banco. Una manera de formalizar esto es exigir la existencia de un algoritmo E que ejecute WITHDRAW u veces con todos los participantes coludidos, extrayendo $u \cdot n$ números seriales $S_1 ... S_{un}$. Se requiere que para todo adversario, la probabilidad de que un banco honesto acepte (S, π) en una operación DEPOSIT, con $S \notin \{S_1 ... S_{un}\}$ es despreciable.

Identificación del doble gasto: asumiendo que $\mathscr{B}, \mathscr{M}_1, \mathscr{M}_2$ son honestos, si $\mathscr{M}_1$ obtiene (S, π_1) de ejecutar SPEND con un adversario, y análogamente $\mathscr{M}_2$ obtiene (S, π_2), se garantiza que IDENTIFY(S, π_1, π_2) emite $(pk_{\mathscr{U}}, \Pi_G)$ tal que VERIFYGUILT$(pk_{\mathscr{U}}, \Pi_G)$ acepta.

Rastreo del doble gasto: si existe una demostración Π_G de que un usuario $\mathscr{U}$ es culpable de doble gasto (tal que VERIFYGUILT acepta), entonces, con alta probabilidad, el procedimiento TRACE$(S, pk_{\mathscr{U}}, \Pi_G, D, n)$ emitirá los números seriales $S_1 ... S_m$ de las monedas de $\mathscr{U}$, con demostraciones $\Pi_1 ... \Pi_m$ de modo que, con alta probabilidad, VERIFYOWNERSHIP$(S_i, \Pi_i, pk_{\mathscr{U}}, n)$ acepta.

Anonimato de los usuarios: se busca que ningún conjunto de usuarios y mercantes, colaborando con el banco, pueda obtener más información sobre las transacciones de un usuario que la deducible de la información ambiental. Más formalmente, se exige que exista un simulador $\mathscr{S}$, que puede utilizar información no disponible para el resto de los actores. Se requiere que $\mathscr{S}$ pueda crear monedas sin tener acceso a billetera alguna, de forma que sean indistinguibles de monedas válidas. Así, $\mathscr{S}$ puede ejecutar la parte del usuario $\mathscr{U}$ en SPEND, sin tener acceso a la billetera W de éste, lo que entrega anonimato a $\mathscr{U}$.

Exculpabilidad: se pide que un usuario honesto no pueda ser culpado de doble gasto. Supongamos se tiene un adversario que ejecuta (tomando el rol de banco) WITHDRAW cualquier número de veces con $\mathscr{U}$, para luego realizar ejecutar SPEND hasta el mismo número de veces con el mismo usuario. Luego, el adversario emite un número serial S y una demostración Π de que S le pertenece a $\mathscr{U}$ y que realizó un doble gasto. Exculpabilidad débil requiere que la probabilidad de que VERIFYOWNERSHIP$(S, pk_{\mathscr{U}}, \Pi, n)$ acepte es despreciable. Para exculpabilidad fuerte, el adversario puede seguir ejecutando SPEND con $\mathscr{U}$, incluso haciéndole gastar dos veces ciertas monedas (en cuyo caso el estado de la billetera se resetea). El adversario emite (S, Π). Se tiene exculpabilidad fuerte si, cuando S no pertenece a $\mathscr{U}$, la probabilidad de que VERIFYOWNERSHIP$(S, pk_{\mathscr{U}}, \Pi, n)$ acepte es despreciable; además, cuando S es un número serial de una moneda que no fue gastada dos veces por $\mathscr{U}$, la probabilidad de que VERIFYGUILT$(S, \Pi, pk_{\mathscr{U}}, n)$ acepte es despreciable.

Una posibilidad es buscar la inclusión de una moneda junto con un mensaje circulante en una red de comunicación anónima. La verificabilidad, de esta forma, sería sobre la validez de la moneda adjunta. Sin embargo, es necesario abordar al menos dos problemas con este enfoque. Por un lado, es necesario que la transferencia de las monedas se realice de forma segura entre los servidores de mezcla. En otras palabras, debe asegurarse que ningún servidor malicioso pueda gastar la moneda (es decir, depositarla en el banco) hasta que el proceso de mezclado haya terminado. También es necesario preservar el anonimato incluso cuando existe transferencia no sólo de mensajes, sino que de monedas.

Las siguientes dos variantes de E-Cash son útiles para tratar estos dos puntos.

3.3.2 E-Cash endosado

Camenisch, Lysyanskaya y Meyerovich definen una variante de E-Cash, E-Cash endosado [17], como una herramienta para realizar transferencias *justas*. En pocas palabras, se posibilita que un mercader pueda entregar el bien o servicio sólo después de obtener una garantía de pago; además, el comprador se asegura de sólo pagar una vez que obtiene estos bienes o servicios. El esquema propuesto por Camenisch, Lysyanskaya y Meyerovich se define de la siguiente forma:

Definición 34 (E-Cash endosado)**.** *Un esquema de E-Cash es un esquema de E-Cash endosado si, en vez de* SPEND*, posee los siguientes algoritmos:*

- SPLITCOIN$(W_j, pk_{\mathscr{B}})$ *permite que un usuario tome una moneda coin de su billetera, y genere* $(\phi, x, y, coin')$*. coin′ es una versión cegada de coin; además, se cumple que* ϕ *es una función homomórfica de un sentido, que* $\phi(x) = y$*, y que la tupla contiene la información necesaria para reconstruir coin.*
- ESPEND$(\mathscr{U}(W, pk_{\mathscr{M}}), \mathscr{M}(sk_{\mathscr{M}}, pk_{\mathscr{B}}, n))$ *permite al usuario entregar una moneda a un mercante. El usuario ejecuta* SPLITCOIN *y entrega* $(\phi, y, coin')$*, quedándose con x. El mercante puede verificar la validez de* $coin'$*.*
- RECONSTRUCT$(\phi, x, y, coin')$*: esta función permite a un mercante reconstruir una moneda, pudiendo ser depositada en el banco sólo si* $\phi(x) = y$*.*

Un esquema de E-Cash endosado permite que un usuario honesto genere hasta $O(2^k)$ monedas cegadas, que no pueden ser rastreadas ni mutuamente enlazadas, a partir de una misma moneda original. Con posterioridad, y dependiendo de la aplicación en particular, puede escoger cuál de estas monedas cegadas *endosar*, permitiendo su depósito en el banco. Las monedas no endosadas pueden ser verificables por el mercante, permitiendo que el intercambio sea correcto. Aplicar este esquema a una red de comunicación anónima permitiría, en potencia, la transferencia justa de monedas entre servidores de mezcla (es decir, que un servidor sólo reciba las monedas una vez realizado el mezclado correcto); sin embargo, no impide el gasto prematuro de estas monedas por parte de servidores corruptos.

3.3.3 E-Cash transferible con un juez

Por otro lado, una variante de E-Cash propuesta por Blazy, Canard, Fuchsbauer, Gouget, Sibert y Traoré [8] es el E-Cash transferible. La principal característica de un esquema de E-Cash transferible es que permite que una moneda pueda ser transferida más de una vez sin tener que ser entregada al banco y retirada entre ambas operaciones. Adicionalmente, se agrega una nueva entidad: un *juez* $\mathscr{J}$ que es quien identifica al autor de un doble gasto, en caso de que el banco así lo detecte.

La definición de un esquema de E-Cash transferible con un juez es similar a la de E-Cash compacto, agregándose el algoritmo $\text{JKEYGEN}(1^k)$, que genera las claves pública y privada del juez, $(pk_{\mathscr{J}}, sk_{\mathscr{J}})$.

Además, IDENTIFY se redefine como $\text{IDENTIFY}(Id, c, c', sk_{\mathscr{J}})$, donde c, c' representan las monedas que potencialmente constituyen el doble gasto; Id representa el identificador repetido en la base de datos del banco. Esto efectivamente separa el rol del banco como participante en retiros y depósitos y el del juez como identificación de usuarios deshonestos.

Sin embargo, Chaum y Pedersen mostraron [23] que una moneda debe crecer en tamaño al ser transferida, lo que limita la utilidad de estos esquemas. Algunas propuestas remedian este problema transfiriendo la información extra a los usuarios (en vez de mantenerla en las monedas) en forma de "recibos" [32]. En el caso de una posible aplicación a redes de comunicación anónima, la opción más factible es la primera, limitándose el ancho de la red de comunicación para que el tamaño de un mensaje no crezca en demasía.

Este último esquema resulta interesante debido a la verificabilidad universal de una propiedad sobre la moneda (en particular, la implementación de E-Cash transferible de [8] utiliza firmas). Sin embargo, tampoco es fácil ver cómo se resuelve la problemática del gasto prematuro por parte de servidores corruptos. De esta forma, analizar las herramientas utilizadas en la construcción de este esquema puede servir en la elaboración de una solución al problema planteado.

Finalmente, este esquema evidencia que una forma básica de tener una propiedad verificable y no falsificable es utilizar firmas.

3.4 Sistema de demostración de Groth-Sahai

Una de las herramientas utilizadas en la instanciación de E-Cash transferible con un juez mostrada en [8] es el sistema de demostración de Groth-Sahai [38]. Éste, de acuerdo a las definiciones del Capítulo 1, es un sistema de demostración de conocimiento no interactivo de testigos indistingui-

bles, y, en casos particulares, puede ser extendido a un sistema de demostración de conocimiento Zero Knowledge.

Si bien existen otros sistemas de demostración con estas características [11], las demostraciones de Groth-Sahai poseen particularidades que las hacen extremadamente atractivas como herramienta criptográfica:

Eficiencia y expresividad: El sistema de demostración propuesto en [11] es capaz de generar demostraciones para cualquier lenguaje en $\mathscr{NP}$, siendo posible, eventualmente, expresar relaciones criptográficas y/o algebraicas. Sin embargo, el proceso de generar demostraciones para estas relaciones resulta en construcciones extremadamente complicadas e ineficientes. En cambio, el sistema de Groth-Sahai está orientado a relaciones que aparecen naturalmente en criptografía, expresándolas de forma mucho más eficiente.

Realeatorizabilidad: La estructura de las demostraciones de Groth-Sahai permite su realeatorización (de manera similar a la realeatorizabilidad de textos cifrados). Esto facilita su adaptación y uso en el contexto de un esquema de redes de comunicación anónimas basados en reencriptación.

Maleabilidad: Las instanciaciones de las demostraciones de Groth-Sahai permiten cierta maleabilidad, haciendo posible obtener demostraciones de enunciados relacionados a partir de una demostración inicial dada.

La definición del sistema de demostración de Groth-Sahai es bastante general; sin embargo, se utilizarán las cuatro situaciones de uso planteadas en el paper original. Dados grupos algebraicos $\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T$, y un *pairing* bilineal $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ entre ellos, los siguientes casos de uso son posibles:

Ecuaciones de productos de *pairings*: El sistema permite demostrar que ciertos valores (entregados como commitments) $X_1 \dots X_n \in \mathbb{G}_1, Y_1 \dots Y_m \in \mathbb{G}_2$ satisfacen ecuaciones de la siguiente forma:

$$\prod_{j=1}^{m} e(A_j, Y_j) \cdot \prod_{i=1}^{n} e(X_i, B_i) \cdot \prod_{i=1}^{n} \prod_{j=1}^{m} e(X_j, Y_i)^{\gamma_{ij}} = t_T$$

con $A_j \in \mathbb{G}_1$, $B_i \in \mathbb{G}_2$, $\gamma_{ij} \in \mathbb{Z}_p$ y $t_T \in \mathbb{G}_T$.

Multiplicación multiescalar en G_1: Se puede demostrar que valores (entregados como commitments) $X_1 \dots X_n \in G_1, y_1 \dots y_m \in \mathbb{Z}_p$ satisfacen ecuaciones de la forma:

$$\prod_{j=1}^{m} A_j^{y_j} \cdot \prod_{i=1}^{n} X_i^{b_i} \cdot \prod_{i=1}^{n} \prod_{j=1}^{m} X_i^{y_j \gamma_{ij}} = t_1$$

con constantes $A_j, t_1 \in \mathbb{G}_1$ y $b_i, \gamma_{ij} \in \mathbb{Z}_p$.

Multiplicación multiescalar en $\mathbb{G}_2$: Dados grupos algebraicos $\mathbb{G}_1, \mathbb{G}_2$, se puede demostrar que valores (entregados como commitments) $x_1 \ldots x_n \in \mathbb{Z}_p, Y_1 \ldots Y_m \in \mathbb{G}_2$ satisfacen ecuaciones de la forma:

$$\prod_{j=1}^{m} Y_j^{a_j} \cdot \prod_{i=1}^{n} B_i^{x_i} \cdot \prod_{i=1}^{n} \prod_{j=1}^{m} Y_j^{x_i \gamma_{ij}} = t_2$$

con constantes $B_i, t_2 \in \mathbb{G}_2$ y $a_j, \gamma_{ij} \in \mathbb{Z}_p$.

Ecuaciones cuadráticas en $\mathbb{Z}_p$: Se puede demostrar que valores (entregados como commitments) $x_1 \ldots x_n \in \mathbb{Z}_n, y_1 \ldots y_m \in \mathbb{Z}_n$ satisfacen ecuaciones de la forma:

$$\sum_{j=1}^{m} a_j y_j + \sum_{i=1}^{n} x_i b_i + \sum_{i=1}^{n} \sum_{j=1}^{m} \gamma_{ij} x_i y_j = t$$

con constantes $a_j, b_i, \gamma_{ij}, t \in \mathbb{Z}p$.

Estos tipos de ecuaciones aparecen comúnmente en esquemas criptográficos, permitiendo la verificabilidad de propiedades en éstos. En el caso de multiplicación multiescalar y ecuaciones cuadráticas, el esquema logra la propiedad de ser perfectamente Zero Knowledge.

El caso de las ecuaciones de productos de *pairings* es más complejo. Para poder tener un esquema perfectamente Zero Knowledge, es necesario poder crear una demostración sin conocer testigo alguno, utilizando parámetros globales. En el caso particular de las ecuaciones de la forma

$$\prod_{j=1}^{m} e(A_j, Y_j) \cdot \prod_{i=1}^{n} e(X_i, B_i) \cdot \prod_{i=1}^{n} \prod_{j=1}^{m} e(X_j, Y_i)^{\gamma_{ij}} = 1_{\mathbb{G}_T}$$

con $A_j \in \mathbb{G}_1$, $B_i \in \mathbb{G}_2$ y $\gamma_{ij} \in \mathbb{Z}_p$ (es decir, ecuaciones de productos de pairings donde $t_T = 1_{\mathbb{G}_T}$), es posible generar esta demostración simulada. Adicionalmente, si la ecuación es de la forma

$$\prod_{j=1}^{m} e(A_j, Y_j) \cdot \prod_{i=1}^{n} e(X_i, B_i) \cdot \prod_{i=1}^{n} \prod_{j=1}^{m} e(X_j, Y_i)^{\gamma_{ij}} = e(A, B)$$

con A, B conocidos, la ecuación puede replantearse como las siguientes dos ecuaciones:

$$Z = A^{-1}$$
$$\prod_{j=1}^{m} e(A_j, Y_j) \cdot \prod_{i=1}^{n} e(X_i, B_i) \cdot \prod_{i=1}^{n} \prod_{j=1}^{m} e(X_j, Y_i)^{\gamma_{ij}} \cdot e(Z, B) = 1_{\mathbb{G}_T}$$

De esta forma, la primera de las ecuaciones es una ecuación de multiplicación multiescalar simple en $\mathbb{G}_1$; la segunda cumple la forma especial de ecuaciones de multiplicación de *pairings* antes descrita, de modo que el sistema de ecuaciones puede ser demostrado de forma perfectamente Zero Knowledge.

Una limitación del sistema de demostración de Groth-Sahai es la imposibilidad de utilizar extracción y simulabilidad (definida a partir de Zero-Knowledge) de manera simultánea. Esto porque el esquema define un parámetro global ω, la cadena de referencia común, que puede ser genera-

do en dos formas: la primera permite la extracción de valores, mientras que la segunda permite la simulación de las demostraciones. Así, este sistema de demostración requiere cuidado en su uso, especialmente a la hora de realizar demostraciones de seguridad. La principal propiedad del sistema es que ningún adversario puede distinguir entre ambos tipos de cadenas de referencia común. Más formalmente, se define el siguiente experimento:

Experimento 8 (Experimento de indistinguibilidad para cadenas de referencia común)**.** $\text{Exp}_{\mathscr{A}}^{\text{Ind-}\omega}(k)$*:*

1. *Se elige un bit al azar* $b \xleftarrow{\$} \{0,1\}$.
2. *Si b es 0, se genera* $\omega \xleftarrow{\$} \text{GENEXTRACT}(k)$. *En caso contrario, se genera* $\omega \xleftarrow{\$} \text{GENSIM}(k)$.
3. *Se entrega* ω *a* $\mathscr{A}$*, que emite un bit* b'.
4. *La salida del experimento se define como 1 si* $b = b'$*, y 0 si no.*

Se define, entonces, la ventaja Ind-ω de $\mathscr{A}$:

$$\text{Adv}_{GS}^{\text{Ind-}\omega}(\mathscr{A}) = |2 \cdot \mathbb{P}[\text{Exp}_{\mathscr{A}}^{\text{Ind-}\omega}(k) = 1] - 1|$$

Donde *GS* es el sistema de Groth-Sahai. Así, una de las propiedades del sistema de demostración de Groth-Sahai es que esta ventaja es pequeña para cualquier adversario. Nótese que cualquiera de las dos formas de generación de la cadena de referencia común pueden utilizarse en una implementación: es al momento de las demostraciones de seguridad que se utiliza una u otra específicamente. Los autores denominan al uso de la cadena de referencia común de extracción la *configuración vinculante*; al uso de la cadena de referencia común de simulación se le denomina la *configuración de ocultamiento*.

A partir del sistema de demostración de Groth-Sahai se han construido diversos algoritmos y protocolos criptográficos. En particular, la verificabilidad de propiedades sin revelar información permite combinar esquemas de firmado y de encriptación de maneras novedosas, abriendo la puerta a nuevas aplicaciones. Así, es útil analizar algunas de estas aplicaciones (especialmente aquellas orientadas a firmas).

3.5 Firmas sobre textos encriptados

Ciertos esquemas criptográficos buscan, utilizando el sistema de demostración de Groth-Sahai, demostrar la presencia de firmas adjuntas a mensajes. Esto constituye una propiedad criptográfica verificable sobre textos cifrados, con lo que puede construirse un esquema que preserve estas propiedades en el contexto de una red de comunicación anónima.

3.5.1 Firmas conmutativas

Las firmas conmutativas (*commuting signatures*) [31] son una herramienta criptográfica que, como su nombre lo indica, combina encriptación y firmado de forma conmutativa.

Definición 35 (Firmas conmutativas)**.** *Un esquema de firmas conmutativas es un esquema que, además de poseer los algoritmos de un esquema de firmas y de encriptación, posee los siguientes algoritmos:*

> SIGCOM*: Dado un commitment C_M a un mensaje M y un par de claves de firma (sk, vk), este algoritmo produce un commitment c_σ a una firma σ sobre M, y una demostración π de que el contenido de c_σ es una firma válida sobre el contenido de C_M.*
>
> ADPRC*: Dado un commitment C_M a un mensaje M, una firma σ sobre M, un commitment c_σ sobre σ, la aleatoriedad ρ_σ usada en este commitment y una demostración π de que σ es una firma válida sobre el contenido de C_M, emite una demostración π de que el contenido de c_σ es una firma válida sobre el contenido de C_M.*
>
> ADPRDC*: De forma inversa al proceso anterior, dado un commitment C_M a un mensaje M, una firma σ sobre M, un commitment c_σ sobre σ, la aleatoriedad ρ_σ usada en este commitment y una demostración π' de que c_σ contiene una firma válida sobre el contenido de C_M, emite una demostración π' de que σ es una firma válida sobre el contenido de C_M.*

Análogamente a los dos algoritmos anteriores (que permiten adaptar una demostración cambiando una firma por su commitment y viceversa), se tienen ADPRC$_M$*,* ADPRDC$_M$*,* ADPRC$_K$ *y* ADPRDC$_K$*. Los dos primeros permiten adaptar una demostración cambiando un mensaje por un commitment sobre éste y viceversa; los dos últimos permiten adaptar una demostración cambiando la clave de verificación de la firma por un commitment sobre ésta y viceversa.*

Con estos algoritmos, una entidad de firmado puede (además de producir firmas sobre textos planos) producir una *firma encriptada* sobre un mensaje encriptado (mediante SIGCOM). Por otro lado, un usuario puede tomar una firma sobre un mensaje y encriptar ambos (usando mecanismos de demostración de Groth-Sahai y los algoritmos de adaptación de demostraciones del esquema), obteniendo una firma encriptada sobre un mensaje encriptado con la misma estructura que el mecanismo anterior. Así, las operaciones de encriptado y firma pueden realizarse en el orden que se prefiera, siendo verificables tanto las firmas sobre mensajes encriptados como aquellas sobre textos planos.

Esto permite, entre otras cosas, firmar textos cifrados sin conocer nada de su contenido, de forma verificable. Además, la implementación propuesta en [31] es realeatorizable, de modo que puede ser utilizada en redes de comunicación anónima basadas en reencriptación.

Sin embargo, el esquema de encriptación utilizado es un esquema de commitments, por lo que la desencriptación no sigue las definiciones estándar de un esquema de encriptación: se realiza revelando la aleatoriedad usada al encriptar el mensaje. Así, no es claro cómo realizar este proceso en el contexto de una red de comunicación anónima.

Además, el espacio de mensajes utilizado en esta implementación no se presta con facilidad para la codificación de mensajes arbitrarios. En particular, se utiliza el espacio de mensajes $\mathbb{M} = \{(G^x, H^x) \mid x \in \mathbb{Z}_p\}$, para G, H fijos, escogidos aleatoriamente en $\mathbb{G}$. Esto se debe al esquema de firmas en el que la construcción se basa, que hace uso de este espacio de mensajes.

3.5.2 Firmas sobre mensajes encriptados realeatorizables

Otro esquema construido en base al sistema de demostración de Groth-Sahai es el de firmas sobre mensajes encriptados realeatorizables [9]. Este esquema también combina encriptación y firmas; sin embargo, y a diferencia de las firmas conmutativas, el esquema de encriptación utilizado se acerca más a la definición estándar de encriptación: así, la desencriptación sólo necesita la clave secreta dk generada junto con la clave pública, no siendo necesario revelar información adicional (como la aleatoriedad utilizada en la encriptación).

Definición 36 (Firmas sobre mensajes encriptados realeatorizables)**.** *Un esquema de firmas sobre mensajes encriptados realeatorizables es un esquema que, además de poseer los algoritmos de un esquema de firmas y de encriptación, posee los siguientes algoritmos:*

$\text{ESIGN}_{sk,pk}(c)$*: Dado un texto cifrado, genera una firma sobre el mensaje encriptado* σ.

$\text{EVERIFY}_{vk,pk}(c,\sigma)$*: Dado un texto encriptado y una firma sobre éste, retorna 1 si la firma es válida, y 0 si no.*

$\text{SIGEXT}_{dk,vk}(c,\sigma)$*: Dada una firma válida sobre un texto encriptado, retorna una firma* σ'.

Se requiere además que los pares (texto cifrado, firma) sean realeatorizables.

Las condiciones de correctitud para un esquema de este tipo son, en adición a aquellas para esquemas de encriptación y firma:

$$\text{EVERIFY}_{vk,pk}(c, \text{ESIGN}_{sk,pk}(c)) = 1$$
$$\text{VERIFY}_{vk}(M, \text{SIGEXT}_{dk,vk}(c, \text{ESIGN}_{sk,pk}(c))) = 1$$

donde $c \xleftarrow{\$} \text{ENCRYPT}_{pk}(M)$, para M arbitrario. Así, el esquema permite generar firmas verificables sobre textos encriptados, que pueden ser convertidas, al momento de la desencriptación del mensaje, en firmas sobre el mensaje original. Además, y de manera similar al esquema anterior, firma

y texto cifrado pueden ser realeatorizados al unísono, lo que posibilita su uso en esquemas basados en realeatorización. Finalmente, el espacio de mensajes utilizado por este esquema es genérico ($\mathbb{M} = \{0,1\}^k$, para k fijo), lo que le da mayores posibilidades de aplicación con respecto a las firmas conmutativas.

Este esquema cumple con la propiedad de ser un esquema de firmas sobre textos cifrados infalsificable, como se encuentra definido en el Capítulo 2.

Una de las desventajas de este esquema es su eficiencia. Debido al esquema de firmas que utiliza, es necesario incluir commitments a cada bit del mensaje a encriptar. Por otro lado, la desencriptación propuesta sólo retorna $\mathrm{F}(M)$, con F el llamado "hash de Waters", donde calcular M a partir de $\mathrm{F}(M)$ toma tiempo exponencial en el largo $|M|$ de M. El autor propone que el esquema se use en casos en que el espacio de mensajes es pequeño, de modo de que invertir F sea factible. Esto dificulta su uso para mensajes genéricos.

La seguridad de un esquema de este tipo se define en base al siguiente experimento:

Experimento 9 (Experimento de infalsificabilidad de firmas sobre textos cifrados)**.** $\text{Sig-forge}^{\text{c-uf}}_{\mathscr{A},\Pi}(k)$*:*

1. *Se ejecuta* SKEYGEN*, obteniéndose las claves pública y privada del esquema de firmas. Se ejecuta n veces* EKEYGEN*, obteniéndose n pares de claves pública y privada de encriptación.*
2. *Se entregan las claves públicas y privadas de encriptación al adversario $\mathscr{A}$, además de la clave de verificación de firmas vk.*
3. *Se proporciona a $\mathscr{A}$ acceso a un oráculo* SIGN*, que recibe un texto cifrado y una de las claves públicas de encriptación, y emite una firma σ sobre éste. Además, internamente, desencripta el texto cifrado y añade el texto plano obtenido a una lista SM de mensajes firmados.*
4. *$\mathscr{A}$ emite un texto cifrado c^*, una clave pública de encriptación pk_j y una firma σ^*.*
5. *Se define m^* como* $\text{DECRYPT}_{dk_j,vk}(c^*)$.
6. *La salida del experimento se define como 1 si $m^* \notin \text{SM}$ y* $\text{EVERIFY}_{vk,pk_j}(c^*,\sigma^*) = 1$*, y 0 en caso contrario.*

Así, se define la infalsificabilidad de firmas sobre textos cifrados:

Definición 37 (Infalsificabilidad de firmas sobre textos cifrados)**.** *Un esquema de firmas sobre mensajes encriptados realeatorizables Π es infalsificable si para todo adversario probabilístico de tiempo polinomial $\mathscr{A}$, existe una función despreciable f tal que:*

$$\mathbb{P}[\text{Sig-forge}^{\text{c-uf}}_{\mathscr{A},\Pi}(k) = 1] \leq f(k)$$

Se denomina a esta probabilidad como la ventaja C-UF ($\text{Adv}^{\text{c-uf}}$*) de A.*

Este esquema se utilizará como base para la solución propuesta, por lo que se mostrará en detalle.

Esquema de Firmas de Waters

Para comenzar, se muestra a continuación el esquema de firmas de Waters [48]. Este esquema es un esquema de firmas como se definió en el Capítulo 1, basado en el problema Diffie-Hellman computacional. Su particularidad consiste en que las ecuaciones de verificación pueden ser expresadas como productos y exponenciaciones de elementos de grupo compatibles con el esquema de demostración de Groth-Sahai. Se especifica el esquema como el Esquema 1.

Esquema 1 Firmas de Waters

function SETUP(1^k)
  $g \xleftarrow{\$} \mathbb{G}$ ▷ $\mathbb{G}$ es un grupo de orden primo $p \approx 2^{2k}$
  $h \xleftarrow{\$} \mathbb{G}$
  $u_0, ..., u_l \xleftarrow{\$} \mathbb{G}$
  return $(\mathbb{G}, h, g, u_0, ..., u_l)$
end function

function KEYGEN(1^k)
  $y \xleftarrow{\$} \mathbb{Z}_p$
  $vk \leftarrow Y = g^y$
  $sk \leftarrow Z = h^y$
  return (vk, sk)
end function

function $\text{SIGN}_{sk=Z}(M)$
  $s \xleftarrow{\$} \mathbb{Z}_p$
  $(\sigma_1, \sigma_2) \leftarrow (Z \cdot \text{F}(M)^s, g^{-s})$
  return $\sigma = (\sigma_1, \sigma_2)$
end function

function $\text{VERIFY}_{vk=Y}(M, \sigma = (\sigma_1, \sigma_2))$
  return $e(g, \sigma_1) \cdot e(\text{F}(M), \sigma_2) = e(Y, h)$
end function

function F(M) ▷ Este es el denominado "hash de Waters"
  $m_1 ... m_l \leftarrow M$ ▷ m_i es el i-ésimo bit de M
  return $u_0 \cdot \prod_{i=1}^{|M|} u_i^{m_i}$
end function

Un problema con el uso de este esquema es la necesidad de codificar la aleatoriedad utilizada en la encriptación para su uso en la demostración de seguridad. Esto implica, por ejemplo, codifi-

car enteros en $\mathbb{Z}_p$ bit por bit, lo que resulta extremadamente costoso. Con el fin de remediar este obstáculo, en [9] se plantea un esquema de firmas modificado, en el cual el mensaje a firmar incluye elementos que contienen la aleatoriedad utilizada para la encriptación, presentado como el Esquema 2.

Encriptación Lineal

Por otro lado, el esquema de encriptación utilizado es el de encriptación lineal [14], basado en el problema lineal decisional. Además de poseer las características de un esquema de encriptación, es homomórfico y realeatorizable [14], como se muestra en el Esquema 3.

Especificación del esquema

Finalmente, encriptación lineal, firmas de Waters modificadas y demostraciones de Groth-Sahai pueden ser conjugados para construir un esquema de firmas sobre mensajes encriptados realeatorizables, como se muestra en el Esquema 4, 5. En este esquema, el algoritmo PROVER corresponde a la generación de una demostración de Groth-Sahai para las siguientes ecuaciones:

$$e(\langle Y^{r_i}\rangle, X_i) = e(c_i, Y), \text{ para } i = 1,2$$

Análogamente, PROVEM corresponde a la generación de demostraciones de Groth-Sahai para la siguiente ecuación:

$$e(c_3, Y) = e(u_0 \prod_{i=1}^{|M|} u_i^{\langle M_i\rangle}, Y) \cdot e(\langle Y^{r_1}\rangle \langle Y^{r_2}\rangle, g)$$

Adicionalmente, GSRANDR y GSRANDM son los respectivos algoritmos de realeatorización de estas demostraciones, según la metodología de Groth-Sahai.

Esquema 2 Firmas de Waters, modificadas

function SETUP(1^k)
  $g \xleftarrow{\$} \mathbb{G}$ ▷ $\mathbb{G}$ es un grupo de orden primo $p \approx 2^{2k}$
  $h \xleftarrow{\$} \mathbb{G}$
  $u_0, ..., u_l \xleftarrow{\$} \mathbb{G}$
  return $(p, \mathbb{G}, \mathbb{G}_T e, g, h, u_0, ..., u_l)$ ▷ e es un *pairing* bilineal no degenerado $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$
end function

function KEYGEN(1^k)
  $y \xleftarrow{\$} \mathbb{Z}_p$
  $vk \leftarrow Y = g^y$
  $sk \leftarrow Z = h^y$
  return (vk, sk)
end function

function $\text{SIGN}_{vk=Y, sk=Z}(M, R_1, R_2, Y_1, Y_2)$
  if $e(R_1, Y) \neq e(g, Y_1)$ **or** $e(R_2, Y) \neq e(g, Y_2)$ **then**
    return $\perp$
  end if
  $s \xleftarrow{\$} \mathbb{Z}_p$
  $(\sigma_1, \sigma_2, \sigma_3, \sigma_4) \leftarrow (Z \cdot (\text{F}(M) \cdot R_1 \cdot R_2)^s, g^{-s}, R_1^{-s}, R_2^{-s})$
  return $\sigma = (\sigma_1, \sigma_2, \sigma_3, \sigma_4)$
end function

function $\text{VERIFY}_{vk=Y}(M, R_1, R_2, Y_1, Y_2, \sigma = (\sigma_1, \sigma_2, \sigma_3, \sigma_4))$
  if $e(R_1, Y) \neq e(g, Y_1)$ **or** $e(R_2, Y) \neq e(g, Y_2)$ **then** ▷ Consistencia de R_i e Y_i
    return $false$
  end if
  if $e(g, \sigma_3) \neq e(\sigma_2, R_1)$ **or** $e(g, \sigma_4) \neq e(\sigma_2, R_2)$ **then** ▷ Correctitud de σ_3, σ_4
    return $false$
  end if
  return $e(g, \sigma_1) \cdot e(\text{F}(M) \cdot R_1 \cdot R_2, \sigma_2) = e(Y, h)$
end function

function F(M)
  $m_1 ... m_l \leftarrow M$
  return $u_0 \cdot \prod_{i=1}^{|M|} u_i^{m_i}$
end function

Esquema 3 Encriptación Lineal

function SETUP(1^k)
  $g \xleftarrow{\$} \mathbb{G}$ ▷ $\mathbb{G}$ es un grupo de orden primo $p \approx 2^k$
  return $(\mathbb{G}, g)$
end function

function KEYGEN(1^k)
  $x_1, x_2 \xleftarrow{\$} \mathbb{Z}_p$
  $dk \leftarrow (x_1, x_2)$
  $pk \leftarrow (X_1, X_2) = (g^{x_1}, g^{x_2})$
  return (pk, dk)
end function

function ENCRYPT$_{pk=(X_1,X_2)}(M)$
  $r_1, r_2 \xleftarrow{\$} \mathbb{Z}_p$
  $c \leftarrow (c_1, c_2, c_3) = (X_1^{r_1}, X_2^{r_2}, g^{r_1+r_2} \cdot M)$
  return c
end function

function DECRYPT$_{dk=(x_1,x_2)}(c = (c_1, c_2, c_3))$
  return $M = c_3 \cdot (c_1^{1/x_1} \cdot c_2^{1/x_2})^{-1}$
end function

function RANDOM$_{pk=(X_1,X_2)}(c = (c_1, c_2, c_3))$
  $r'_1, r'_2 \xleftarrow{\$} \mathbb{Z}_p$
  $c' \leftarrow (c_1 \cdot X_1^{r'_1}, c_2 \cdot X_2^{r'_2}, g^{r'_1+r'_2} \cdot c_3)$
  return c'
end function

Esquema 4 Firmas sobre mensajes encriptados realeatorizables

function SETUP(1^k)
  $(p, \mathbb{G}, \mathbb{G}_T, e, g) \xleftarrow{\$}$ GENGROUP(1^k) ▷ Definición de los grupos bilineales
  $h \xleftarrow{\$} \mathbb{G}$
  $u_0, ..., u_l \xleftarrow{\$} \mathbb{G}$ ▷ l es el largo de los mensajes a utilizar
  $u \leftarrow (u_0, ..., u_l)$
  return $(p, \mathbb{G}, \mathbb{G}_T, e, g, h, u)$
end function

function EKEYGEN(1^k)
  $x_1, x_2 \xleftarrow{\$} \mathbb{Z}_p$
  $dk \leftarrow (x_1, x_2)$
  $pk \leftarrow (X_1, X_2) = (g^{x_1}, g^{x_2})$
  return (pk, dk)
end function

function ENCRYPT$_{pk=(X_1,X_2),vk=Y}(M)$
  $r_1, r_2 \xleftarrow{\$} \mathbb{Z}_p$
  $c \leftarrow (c_1, c_2, c_3) = (X_1^{r_1}, X_2^{r_2}, g^{r_1+r_2} \cdot \mathrm{F}(M))$
  $\Pi_r \leftarrow$ PROVER(Y, c, r_1, r_2)
  $\Pi_M \leftarrow$ PROVEM(Y, c, r_1, r_2, M)
  return $C = (c, \Pi_r, \Pi_M)$
end function

function DECRYPT$_{dk=(x_1,x_2)}(C = (c, \Pi_r, \Pi_M), \sigma))$
  if SIGVERIFY$_{vk}(C, \sigma) \neq true$ **then**
    return $\perp$
  end if
  $F \leftarrow c_3 \cdot (c_1^{1/x_1} \cdot c_2^{1/x_2})^{-1}$ ▷ $F = \mathrm{F}(M)$
  return $M = \mathrm{F}^{-1}(F)$ ▷ Esto es posible si el espacio de mensajes $\mathbb{M}$ es pequeño.
end function

Esquema 5 Firmas sobre mensajes encriptados realeatorizables (continuación)

function SKEYGEN(1^k)
$y \xleftarrow{\$} \mathbb{Z}_p$
$vk \leftarrow Y = g^y$
$sk \leftarrow Z = h^y$
return (vk, sk)
end function

function ESIGN$_{sk=Z,pk=(X_1,X_2)}(C = (c, \Pi_r, \Pi_M))$
if (VERIFYPROOFR(c, Π_r) **and** VERIFYPROOFM$(c, \Pi_M)) \neq true$ **then**
return $\perp$
end if
$s \xleftarrow{\$} \mathbb{Z}_p$
return $\sigma = (c_1^s, c_2^s, Z \cdot c_3^s, c_1^s, c_2^s, g^s)$
end function

function EVERIFY$_{vk=Y,pk=(X_1,X_2)}(C = (c, \Pi_r, \Pi_M), \sigma = (\sigma_1, \sigma_2, \sigma_3, \sigma_4, \sigma_5, \sigma_6))$
if VERIFYPROOFS$(c, \Pi_r, \Pi_M) \neq true$ **then**
return $false$
end if
if $e(\sigma_1, X_1) \neq e(c_1, \sigma_4)$ **or** $e(\sigma_1, g) \neq e(c_1, \sigma_6)$ **then**
return $false$
end if
if $e(\sigma_2, X_2) \neq e(c_2, \sigma_5)$ **or** $e(\sigma_2, g) \neq e(c_2, \sigma_6)$ **then**
return $false$
end if
return $e(\sigma_3, g) = e(h, Y) \cdot e(c_3, \sigma_6)$
end function

function RANDOM$_{vk=Y,pk=(X_1,X_2)}(C = (c, \Pi_r, \Pi_M), \sigma = (\sigma_1, \sigma_2, \sigma_3, \sigma_4, \sigma_5, \sigma_6))$
$r'_1, r'_2, s' \xleftarrow{\$} \mathbb{Z}_p$
$\sigma' \leftarrow (\sigma_1 \cdot c_1^{s'} \cdot \sigma_4^{r'_1} \cdot X_1^{r'_1 s'}, \sigma_2 \cdot c_2^{s'} \cdot \sigma_5^{r'_2} \cdot X_2^{r'_2 s'}, \sigma_3 \cdot c_3^{s'} \cdot \sigma_6^{r'_1 + r'_2} \cdot g^{(r'_1 + r'_2)s'}, \sigma_4 \cdot X_1^{s'}, \sigma_5 \cdot X_2^{s'}, \sigma_6 \cdot g^{s'})$
$c' \leftarrow (c_1 \cdot X_1^{r'_1}, c_2 \cdot X_2^{r'_2}, c_3 \cdot g^{r'_1 + r'_2})$
$\Pi'_r, \Pi'_M \leftarrow \text{GSRANDR}(\Pi_r, r_1, r_2), \text{GSRANDM}(\Pi_M)$
return $(C' = (c', \Pi'_r, \Pi'_M), \sigma')$
end function

function F(M)
$m_1 ... m_l \leftarrow M$
return $u_0 \cdot \prod_{i=1}^{|M|} u_i^{m_i}$
end function

Capítulo 4

Descripción de la Solución

El esquema propuesto se basa en la construcción de una red de mezcla expuesta en [18]. Sin embargo, se utiliza un esquema de encriptación basado en el esquema de firmas sobre mensajes aleatorizables expuesto en [9].

Se consideran las siguientes entidades participantes de la red de mezcla:

Usuarios: Existen n usuarios $U_1, ..., U_t$, que hacen uso de la red.

Servidores de mixing: Existen m servidores $M_1, ..., M_m$ que realizan el mezclado de los mensajes.

Servidores de certificación: Existen k servidores de certificación $S_1, ...S_s$ que pueden emitir firmas (cada uno con claves de verificación distintas).

Autoridad confiable: Existe una autoridad confiable $\mathscr{K}$ que verifica la correcta ejecución de los servidores de mezcla. Esto lo realiza participando como contraparte en las demostraciones interactivas de correctitud de funcionamiento (por ejemplo, la demostración de mezclado correcto). Participa también en la configuración inicial del sistema.

El problema de la desencriptación distribuida de mensajes no se aborda en esta solución, considerándose que una autoridad confiable realiza la desencriptación de los mensajes luego de realizarse el proceso de mezcla.

Se detalla a continuación el protocolo de mezclado.

4.1 Generación de claves

El protocolo de generación de claves KEYGEN se compone de dos partes. Por un lado, se generan las claves del esquema de encriptación umbral lineal subyacente; por otro, se generan las claves del esquema de firma para cada autoridad de certificación, utilizando los algoritmos EKEYGEN y SKEYGEN. Se publican las claves públicas del esquema de encriptación y de cada autoridad firmante.

Cada autoridad firmante debe realizar, además, una demostración de conocimiento de su clave privada, consistente en una demostración de Groth-Sahai del firmado correcto de un mensaje escogido por la autoridad confiable $\mathscr{K}$.

4.2 Encriptación y autorización de mensajes

La sesión de mezclado comienza cuando la autoridad publica un mensaje de inicio, junto con un identificador de sesión. Cada servidor de mezcla decide la identidad de las autoridades de certificación cuyas firmas requerirá en los mensajes que procese, publicándolas. En cada sesión, cada usuario envía un único mensaje encriptado utilizando el protocolo de encriptación ENCRYPT, que consiste en:

- Encriptar el mensaje utilizando el algoritmo ENCRYPT del esquema de encriptación de la red, obteniendo c.
- Solicitar una firma σ sobre c a una autoridad certificadora de su elección.
- Realeatorizar el par $C = (c, \sigma)$, obteniendo $C' = (c', \sigma')$.
- Ejecutar una demostración de conocimiento de su mensaje con la autoridad confiable $\mathscr{K}$.

Nótese que el proceso de petición de firmas puede repetirse tantas veces como sea necesario con diferentes autoridades de certificación, siempre y cuando se realice la realeatorización del texto cifrado entre cada proceso (evitando comprometer así el anonimato).

De acuerdo a lo especificado en el Capítulo 3, un usuario honesto debe poder tener la garantía de que su mensaje no será descartado si cumple las condiciones de filtrado. En este caso, las condiciones de filtrado no se especifican, sino que están simbolizadas por la presencia o no de las firmas necesarias. Una instanciación más específica de este esquema pudiera requerir que un usuario demuestre (interactivamente) una propiedad de su texto cifrado, obteniendo una firma sólo si esta demostración es exitosa.

Cabe mencionar que este enfoque cobra sentido al existir conceptos como la búsqueda de palabras clave en textos cifrados por parte de terceros [39] [4]: diferentes autoridades de certificación pudieran utilizar mecanismos de este tipo para justificar la entrega o no de su firma. La correcta ejecución de estos protocolos debe ser públicamente verificable, de modo de eliminar del sistema a aquellas autoridades certificadoras que se comporten inapropiadamente. En este caso, se considera que todo usuario que solicite una firma debiera ser capaz de obtenerla. En caso contrario, se puede denunciar la negativa de la autoridad certificadora, con la que se la remueve del sistema de forma permanente.

La demostración de conocimiento del texto plano subyacente es una demostración interactiva, que consiste en los siguientes pasos:

- $\mathscr{K}$ escoge una clave pública pk para encriptación lineal, que envía al usuario $\mathscr{U}$.
- El usuario encripta el mismo mensaje bajo pk, generando c_2 y realiza una demostración de que c y c_2 poseen el mismo texto plano subyacente. Esta demostración puede realizarse siguiendo la metodología de Groth-Sahai.
- La demostración es aceptada si esta última demostración es correcta.

Esta demostración posee extraibilidad (el extractor sólo debe jugar el rol de la autoridad confiable y generar la clave de desencriptación dk junto a la clave pk). De esta forma, puede extraer el texto plano subyacente desencriptando c_2. Por otro lado, la demostración posee la propiedad de Zero-Knowledge: basta con utilizar la propiedad de Zero-Knowledge de la demostración de Groth-Sahai involucrada para obtener una demostración válida sin conocer testigo alguno. Es interesante notar que ambas propiedades pueden tenerse de forma simultánea, lo que se utilizará para la demostración de seguridad.

Una vez que la demostración es aceptada, se desecha c_2 y la demostración de igualdad; $\mathscr{K}$ ingresa cada texto cifrado válido c_i correspondiente a un usuario U_i como C_i^0 en una lista L_0 de textos cifrados aceptados. La lista de textos planos recibidos correctamente $K = \{m_1, ..., m_{t_0}\}$ es la entrada de la red de mezcla para una sesión dada.

4.3 Ejecución de la red

El protocolo de ejecución EXEC de la red realiza el mezclado y la desencriptación de los mensajes.

Mezclado

Luego de que L_0 se publica, los servidores de mezcla realizan esta operación, uno por uno. El i-ésimo servidor de mezcla lee una lista $L_{i-1} = \{C_1^{i-1}, ..., C_{t_{i-1}}^{i-1}\}$ ya publicada. Nótese que, en el caso general, $t_i \leq t$, debido a que mensajes pudieran ser descartados debido a que no cumplen con la política de un cierto servidor de mezcla, o en el caso en que la demostración de conocimiento de texto plano correspondiente no haya sido realizada de forma correcta.

A continuación, M_i verifica las firmas contenidas en cada texto cifrado según su propio protocolo, descartándose aquellos para los que la verificación falle. Nótese que esta verificación puede ser hecha en todo momento y por cualquier entidad, lo que obliga a los servidores de mezcla a realizar el filtrado de manera correcta. Para los mensajes no descartados, se escoge una permutación aleatoria π y forma una lista $L_i = \{C_1^i, ..., C_{t_i}^i\}$, donde:

$$C_j^i \xleftarrow{\$} \text{RANDOM}_{vk,pk}(c_j^i)$$

Nótese que en este punto se c_j^i contiene todas las firmas adjuntas; en otras palabras, se realeatoriza el texto cifrado y todas las firmas adjuntas a éste. Luego de este proceso de realeatorización, M_i publica L_i. Además, realiza una demostración con la autoridad $\mathscr{K}$ de que realizó el mezclado de manera correcta, así como el descarte de aquellos mensajes sin una firma apropiada. Se denota PROVE a esta demostración. Si $\mathscr{K}$ no acepta esta demostración, se descarta L_i, definiéndose $L_i = L_{i-1}$, y se anuncia que el servidor M_i es deshonesto. De la misma forma, si el filtrado de mensajes no se realiza de manera correcta, se declara el servidor de mezcla como deshonesto.

Desencriptación

Luego de que el último servidor de mezcla M_m emite una lista $\bar{L} = \{\bar{C}_1, ..., \bar{C}_{t_m}\}$, se procede a la desencriptación de cada mensaje.

En un esquema de mezclado usual, es necesario que este procedimiento se realice a través de un conjunto de servidores de desencriptación D. En esta instanciación, como ya se mencionó, no se aborda este problema. Así, se considera que una autoridad confiable es la que lleva a cabo este proceso, computando, a partir de la lista $\bar{L}$, una lista de textos planos $\bar{K} = \{\bar{m}_1, ..., \bar{m}_{t_m}\}$, que publica. Se denomina $\bar{K}$ como la salida de la red de mezcla.

El detalle de los algoritmos de generación de claves, encriptación y firma se especifica en el Apéndice.

Capítulo 5

Validación de la Solución

Con el objetivo de validar la solución propuesta, se comenzará por mostrar que se ajusta a la definición del problema expuesta en el Capítulo 2. En otras palabras, se demostrará la seguridad del esquema planteado. El principal resultado que valida la seguridad del esquema construido es el siguiente:

Teorema 3. *El esquema propuesto corresponde a una red de mezcla con verificación de mensajes certificados segura ante adversarios que comprometen cualquier número de usuarios, cualquier número de autoridades certificadoras, y a lo más* $m-1$ *servidores de mezcla.*

Para demostrar este resultado, y de la definición de seguridad para una red de mezcla de mensajes certificados (Cap. 2), debe mostrarse que, para todo adversario $\mathscr{A}$ existen funciones despreciables f, g tales que:

$$\begin{aligned} \mathrm{Adv}^{\text{MIX-ANON}}_{\mathscr{M}\mathscr{I}\mathscr{X}}(\mathscr{A}) &\leq f(k) \\ \mathrm{Adv}^{\text{MIX-UF}}_{\mathscr{M}\mathscr{I}\mathscr{X}}(\mathscr{A}) &\leq g(k) \end{aligned}$$

A continuación se demostrarán estas dos relaciones por separado.

5.1 Anonimato del esquema

Se busca demostrar el siguiente lema:

Lema 1. *Para cualquier adversario de tiempo polinomial* $\mathscr{A}$ *que compromete cualquier número de servidores de certificación, cualquier número de usuarios y a lo más* $m-1$ *servidores de mezcla, la ventaja* $\mathrm{Adv}^{\text{MIX-ANON}}_{\mathscr{M}\mathscr{I}\mathscr{X}}(\mathscr{A})$ *es pequeña.*

Demostración. El anonimato del esquema puede medirse ofreciendo una cota para el término $\mathrm{Adv}^{\text{MIX-ANON}}_{\mathscr{MIX}}(\mathscr{A})$. Esto puede llevarse a cabo mediante una sucesión de experimentos. Esta técnica es común en criptografía, siendo detallada en [45] y [6]; además, existen esfuerzos por automatizar este tipo de demostraciones [7]. Sin pérdida de generalidad, se asume que existe sólo un servidor de mezcla honesto.

1. El primer experimento es idéntico al experimento ANON-UF.
2. Este experimento es idéntico al anterior, salvo que el parámetro global ω utilizado para las demostraciones de conocimiento de los textos planos es el de simulación, ω_{sim}, en vez de ω_{crs}.
3. Este experimento es idéntico al anterior, salvo que el parámetro global ω^s utilizado para las demostraciones de conocimiento de los textos planos es el de extracción, ω^s_{ext}, en vez de ω^s_{crs}.
4. Este experimento se diferencia del anterior en que las demostraciones de conocimiento del texto plano son realizadas utilizando la función de simulación S en vez del algoritmo de demostración P.
5. Este experimento se diferencia del anterior en que el servidor de mezcla honesto, en vez de mezclar la entrada que recibe, toma los textos planos originales (incluyendo los extraídos de los textos cifrados emitidos por usuarios corruptos) y los encripta. A continuación, elige una permutación aleatoria π y la aplica a esta nueva lista de textos cifrados. Presenta esta lista como la salida de la operación de mezclado, realizando la simulación de la demostración de mezclado correcto.
6. Este experimento se diferencia del anterior en que no se utiliza la clave de desencriptación. En vez de desencriptar cada texto cifrado luego del mezclado, se extrae la permutación total aplicada a los textos cifrados aplicando la extracción de testigo para las demostraciones de mezclado correcto. A continuación, se aplica esta permutación a la lista de textos planos originales (se obtiene el texto plano para los usuarios corruptos utilizando la extraibilidad de la demostración de conocimiento). Se presenta esta permutación como la salida de la red de mezcla.
7. Este experimento se diferencia del anterior en que, en vez de desencriptar los textos cifrados emitidos por el proceso total de mezclado, se escoge una permutación aleatoria de los textos originales y se presenta como la salida de la red.
8. Este experimento se diferencia del anterior en que, en vez de utilizar encriptaciones de los textos de los usuarios honestos, se encriptan copias del mensaje $M^* = (1, ..., 1)$.

La demostración de seguridad consistirá, entonces, en relacionar los comportamientos de un adversario $\mathscr{A}$.

Para cada experimento i, se define su salida como 1 si el bit b' emitido por $\mathscr{A}$ es igual al bit b escogido aleatoriamente al principio de cada experimento. Finalmente, se define para cada

experimento la ventaja de $\mathscr{A}$ como:

$$\mathrm{Adv}_{\mathrm{Exp}^i}(\mathscr{A}) = |2 \cdot \mathbb{P}[\mathrm{Exp}^i_{\mathscr{A}}(k) = 1] - 1|$$

Experimentos 1 y 2

Por definición, la ventaja de $\mathscr{A}$ en el Experimento 1 es exactamente la ventaja MIX-ANON de $\mathscr{A}$. Por otro lado, se puede plantear un adversario, explicitado en el algoritmo 6, que distingue entre ω_{crs} y ω_{sim}.

Algoritmo 6 Adversario $\mathscr{B}^1$

function $\mathscr{B}^1(\omega)$
  $b^* \xleftarrow{\$} \{0,1\}$
  $b_A \xleftarrow{\$} \mathrm{Exp}^{1,2}(b^*, \omega, \mathscr{A})$
  $b_B \leftarrow b_A = b^*$
  return b_B
end function

En este adversario, $\mathrm{Exp}^{1,2}$ consiste en ejecutar el experimento ANON-UF utilizando el parámetro ω dado en vez de generarlo, y el bit b^* al momento de escoger la permutación de los mensajes de los usuarios honestos a utilizar. Es claro que si ω es generado , este experimento es equivalente al Experimento 1; por otro lado, si ω es generado mediante la función de simulación S_1, se tiene el Experimento 2. La ventaja de $\mathscr{B}^1$ ante el experimento de distinción de ω (definido en el experimento 8), Ind-ω, se define como:

$$\mathrm{Adv}^{\text{Ind-}\omega}_{GS}(\mathscr{B}^1) = |2 \cdot \mathbb{P}[\mathrm{Exp}^{\text{Ind-}\omega}_{\mathscr{B}^1}(k) = 1] - 1|$$

donde el valor del bit b_ω es elegido aleatoriamente; el valor de ω entregado al adversario $\mathscr{B}^1$ es generado normalmente si $b_\omega = 0$; en cambio, es generado mediante la función de simulación S_1 si $b_\omega = 1$. Por la definición del experimento Ind-ω, y condicionando sobre el bit b_B emitido por $\mathscr{B}^1$, se tiene:

$$\begin{aligned}
\mathbb{P}[\mathrm{Exp}^{\text{Ind-}\omega}_{\mathscr{B}^1}(k) = 1] &= \mathbb{P}[b_B = b_\omega] \\
&= \mathbb{P}[b_B = b_\omega | b_\omega = 1] \cdot \mathbb{P}[b_\omega = 1] + \mathbb{P}[b_B = b_\omega | b_\omega = 0] \cdot \mathbb{P}[b_\omega = 0] \\
&= \frac{1}{2}\mathbb{P}[b_A = b^* | b_\omega = 1] + \frac{1}{2}\mathbb{P}[b_A \neq b^* | b_\omega = 0] \\
&= \frac{1}{2}(1 + \mathbb{P}[b_A = b^* | b_\omega = 1] - \mathbb{P}[b_A = b^* | b_\omega = 0])
\end{aligned}$$

Luego:

$$\begin{aligned}
2\cdot(2\cdot\mathbb{P}[\mathrm{Exp}^{\text{Ind-}\omega}_{\mathscr{B}^1}(k)=1]-1) =& (2\cdot\mathbb{P}[b_A=b^*|b_\omega=1]-1)-\\
& (2\cdot\mathbb{P}[b_A=b^*|b_\omega=0]-1)\\
\implies (2\cdot\mathbb{P}[b_A=b^*|b_\omega=0]-1) =& (2\cdot\mathbb{P}[b_A=b^*|b_\omega=1]-1)-\\
& 2\cdot(2\cdot\mathbb{P}[\mathrm{Exp}^{\text{Ind-}\omega}_{\mathscr{B}^1}(k)=1]-1)\\
\implies |2\cdot\mathbb{P}[b_A=b^*|b_\omega=0]-1| =& |(2\cdot\mathbb{P}[b_A=b^*|b_\omega=1]-1)-\\
& 2\cdot(2\cdot\mathbb{P}[\mathrm{Exp}^{\text{Ind-}\omega}_{\mathscr{B}^1}(k)=1]-1)|\\
\implies |2\cdot\mathbb{P}[b_A=b^*|b_\omega=0]-1| \leq& |2\cdot\mathbb{P}[b_A=b^*|b_\omega=1]-1|+\\
& 2\cdot|2\cdot\mathbb{P}[\mathrm{Exp}^{\text{Ind-}\omega}_{\mathscr{B}^1}(k)=1]-1|\\
\implies \mathrm{Adv}_{\mathrm{Exp}^1}(\mathscr{A}) \leq& \mathrm{Adv}_{\mathrm{Exp}^2}(\mathscr{A})+2\cdot\mathrm{Adv}^{\text{Ind-}\omega}_{GS}(\mathscr{B}^1)
\end{aligned}$$

Nótese que esto asume que en el Experimento 1 se utilizaba la configuración vinculante. Si el Experimento 1 utilizara la configuración de ocultación, los experimentos 1 y 2 no poseen diferencia alguna, con lo que $\mathrm{Adv}_{\mathrm{Exp}^1}(\mathscr{A}) = \mathrm{Adv}_{\mathrm{Exp}^2}(\mathscr{A})$. De cualquier manera, esta cota dada cubre ambos casos (pues la ventaja de $\mathscr{B}^1$ siempre es mayor o igual a cero).

Experimentos 2 y 3

El paso entre estos dos experimentos es análogo al paso anterior. Así, puede plantearse un adversario $\mathscr{B}^2$ para el experimento de distinción Ind-ω, alterándose el parámetro ω^s en este caso. De esta forma, y análogamente al caso de los experimentos 1 y 2, se tiene:

$$\mathrm{Adv}_{\mathrm{Exp}^2}(\mathscr{A}) \leq \mathrm{Adv}_{\mathrm{Exp}^3}(\mathscr{A}) + 2\cdot\mathrm{Adv}^{\text{Ind-}\omega}_{GS}(\mathscr{B}^2)$$

Experimentos 3 y 4

La diferencia entre los experimentos 3 y 4 consiste en que las demostraciones de conocimiento de texto plano realizadas por parte de los usuarios honestos son simuladas, en vez de ser construidas mediante el protocolo. Sin embargo, en el sistema de demostración de Groth-Sahai las demostraciones simuladas siguen la misma distribución que las demostraciones honestas (dado que posee la propiedad de Zero-Knowledge componible). Esto implica que

$$\mathrm{Adv}_{\mathrm{Exp}^3}(\mathscr{A}) = \mathrm{Adv}_{\mathrm{Exp}^4}(\mathscr{A})$$

Experimentos 4 y 5

La diferencia entre estos dos experimentos consiste en que el servidor de mezcla honesto elige una permutación aleatoria de nuevos textos cifrados y demuestra el mezclado correcto utilizando el proceso de simulación del esquema de demostración. Es útil notar que la lista de textos cifrados generada por este servidor distribuye de manera idéntica en ambos experimentos: la composición de una permutación aleatoria con una permutación fija posee la misma distribución que una permutación aleatoria; la realeatorización del texto cifrado sigue la misma distribución que un texto cifrado original. De esta forma, la única diferencia apreciable entre ambos experimentos es la diferencia entre las demostraciones.

De esta forma, se puede plantear un adversario, explicitado en 7, que distingue entre demostraciones correctas y simuladas.

Algoritmo 7 Adversario $\mathscr{B}^3$

function $\mathscr{B}^3(\omega)$
  $b^* \xleftarrow{\$} \{0,1\}$
  $b_A \xleftarrow{\$} \text{Exp}^{4,5}(b^*, \text{SHUFFLEPROVE}, \mathscr{A})$
  $b_B \leftarrow b_A = b^*$
  return b_B
end function
function SHUFFLEPROVE$(\pi, \vec{r}, \vec{C}, \vec{C'})$
  $r \xleftarrow{\$} \mathbb{R}^t$ ▷ Se escogen los valores aleatorios a usar en el protocolo
  return PROVE$(\pi, r, \vec{C}, \vec{C'})$
end function

En este adversario, $\text{Exp}^{4,5}$ consiste en llevar a cabo el Experimento 3 (o 4, de forma equivalente) utilizando el bit b^* para escoger la permutación de los mensajes de los usuarios honestos. Además, el algoritmo PROVE consiste en realizar la demostración de mezclado con el oráculo (que puede estar realizando una demostración correcta o simulada).

Como la extraibilidad de las demostraciones de Groth-Sahai es perfecta, se puede realizar la reproducción de las firmas que tenía cada texto cifrado, ya que se dispone de las claves privadas de los servidores de firma. Adicionalmente, se sabe exactamente qué firmas generar debido a que los servidores deben realizar el filtrado de forma correcta (o son eliminados).

La función SHUFFLEPROVE se utiliza en el único servidor de mezcla honesto. Este servidor honesto entrega la permutación y los vectores de textos cifrados de entrada y salida, además de la aleatoriedad usada en la realeatorización de éstos, al oráculo de demostración. Este oráculo entrega una traza de la interacción entre el servidor y la autoridad $\mathscr{K}$ con la cual se realiza la demostración. $\mathscr{B}^3$ publica esta traza como la generada por la demostración de mezclado del servidor. Es evidente que en el caso en que el oráculo PROVE emita demostraciones correctas, $\mathscr{A}$ se encuentra ante el Experimento 4; por otro lado, si el oráculo emite demostraciones simuladas, el adversario $\mathscr{A}$

se encuentra en el Experimento 5. Así, de la definición de la propiedad de Zero Knowledge con verificador honesto:

$$\begin{aligned}\mathbb{P}[\mathrm{Exp}^{\text{Ind-zk}}_{\mathscr{B}^3}(k)=1] &= \mathbb{P}[b_B = b_{zk}] \\ &= \mathbb{P}[b_B = b_{zk} | b_{zk}=1]\cdot\mathbb{P}[b_{zk}=1] + \mathbb{P}[b_B = b_{zk}|b_{zk}=0]\cdot\mathbb{P}[b_{zk}=0] \\ &= \frac{1}{2}\mathbb{P}[b_A = b^* | b_{zk}=1] + \frac{1}{2}\mathbb{P}[b_A \neq b^* | b_{zk}=0] \\ &= \frac{1}{2}(1+\mathbb{P}[b_A = b^*|b_{zk}=1] - \mathbb{P}[b_A = b^*|b_{zk}=0])\end{aligned}$$

Luego:

$$\begin{aligned}2\cdot(2\cdot\mathbb{P}[\mathrm{Exp}^{\text{Ind-}zk}_{\mathscr{B}^3}(k)=1]-1) =& (2\cdot\mathbb{P}[b_A=b^*|b_{zk}=1]-1) - \\ & (2\cdot\mathbb{P}[b_A=b^*|b_{zk}=0]-1) \\ \implies (2\cdot\mathbb{P}[b_A=b^*|b_{zk}=0]-1) =& (2\cdot\mathbb{P}[b_A=b^*|b_{zk}=1]-1) - \\ & 2\cdot(2\cdot\mathbb{P}[\mathrm{Exp}^{\text{Ind-}zk}_{\mathscr{B}^3}(k)=1]-1) \\ \implies |2\cdot\mathbb{P}[b_A=b^*|b_{zk}=0]-1| =& |(2\cdot\mathbb{P}[b_A=b^*|b_{zk}=1]-1) - \\ & 2\cdot(2\cdot\mathbb{P}[\mathrm{Exp}^{\text{Ind-}zk}_{\mathscr{B}^3}(k)=1]-1)| \\ \implies |2\cdot\mathbb{P}[b_A=b^*|b_{zk}=0]-1| \leq& |2\cdot\mathbb{P}[b_A=b^*|b_{zk}=1]-1| + \\ & 2\cdot|2\cdot\mathbb{P}[\mathrm{Exp}^{\text{Ind-}zk}_{\mathscr{B}^3}(k)=1]-1| \\ \implies \mathrm{Adv}_{\mathrm{Exp}^4}(\mathscr{A}) \leq& \mathrm{Adv}_{\mathrm{Exp}^5}(\mathscr{A}) + 2\cdot\mathrm{Adv}^{\text{Ind-zk}}(\mathscr{B}^3)\end{aligned}$$

donde $\mathrm{Adv}^{\text{Ind-zk}}$ corresponde a la ventaja especificada en la sección 1.3.7, definición 25,.

Experimentos 5 y 6

Para comenzar, es útil analizar el mecanismo de extracción de la demostración de conocimiento del texto plano utilizado. Para extraer el texto plano, basta con que el verificador honesto genere la clave privada *dk* correspondiente a la clave pública entregada al adversario durante la demostración, con la que puede desencriptar la encriptación entregada por el adversario, obteniendo el texto plano. Como el sistema de demostración de igualdad de textos planos utilizado (una demostración de Groth-Sahai) posee consistencia perfecta, la extraibilidad resulta ser perfecta. Por esto, el hecho de extraer en vez de desencriptar no altera el comportamiento del adversario en lo absoluto.

Queda por argumentar que la extracción de las permutaciones utilizadas se realiza de forma correcta e indistinguible. Efectivamente, la propiedad de emulación extendida con testigo (que el esquema de demostración de mezclado correcto posee) postula que la utilización del extractor de testigos es computacionalmente indistinguible. Se plantea así el adversario $\mathscr{B}^4$, explicitado en el algoritmo 8.

Algoritmo 8 Adversario $\mathscr{B}^4$

function $\mathscr{B}^4(\omega)$
  $b^* \xleftarrow{\$} \{0,1\}$
  $b_A \xleftarrow{\$} \text{Exp}^{1,2}(b^*, \omega, \mathscr{A})$
  $b_B \leftarrow b_A = b^*$
  return b_B
end function
function VERIFY$(x, \mathscr{A})$
  $tr \xleftarrow{\$}$ GETTRACE$(x, s(\mathscr{A})$
  return tr
end function

En este adversario, $\text{Exp}^{5,6}$ consiste en llevar a cabo el Experimento 5 (ó 6, de forma equivalente) utilizando el bit b^* para escoger la permutación de los mensajes de los usuarios honestos. Nótese que $\mathscr{B}$ puede desencriptar correctamente los mensajes, por lo que no necesita obtener un testigo a partir del emulador. La función VERIFY se utiliza para realizar la demostración con $\mathscr{A}$. La expresión $s(\mathscr{A})$ simboliza la entrega del estado de $\mathscr{A}$ al oráculo. El adversario $\mathscr{B}^4$ utiliza esta función para realizar las demostraciones de mezclado correcto con los servidores comprometidos (realizando el rebobinado si fuera necesario).

Es claro que si la traza del oráculo se obtuvo de una ejecución correcta del protocolo de demostración, $\mathscr{A}$ observa lo correspondiente al Experimento 5. Por otro lado, si la traza del oráculo corresponde a la emitida por un emulador, lo que observa $\mathscr{A}$ corresponde al Experimento 6.

$$\begin{aligned}
\mathbb{P}[\text{Exp}^{\text{Ind-tr}}_{\mathscr{B}^4}(k) = 1] &= \mathbb{P}[b_B = b_{tr}] \\
&= \mathbb{P}[b_B = b_{tr} | b_{tr} = 1] \cdot \mathbb{P}[b_{tr} = 1] + \mathbb{P}[b_B = b_{tr} | b_{tr} = 0] \cdot \mathbb{P}[b_{tr} = 0] \\
&= \frac{1}{2}\mathbb{P}[b_A = b^* | b_{tr} = 1] + \frac{1}{2}\mathbb{P}[b_A \neq b^* | b_{tr} = 0] \\
&= \frac{1}{2}(1 + \mathbb{P}[b_A = b^* | b_{tr} = 1] - \mathbb{P}[b_A = b^* | b_{tr} = 0])
\end{aligned}$$

Luego:

$$\begin{aligned}
2 \cdot (2 \cdot \mathbb{P}[\text{Exp}^{\text{Ind-}tr}_{\mathscr{B}^4}(k) = 1] - 1) =& (2 \cdot \mathbb{P}[b_A = b^* | b_{zk} = 1] - 1) - \\
& (2 \cdot \mathbb{P}[b_A = b^* | b_{tr} = 0] - 1) \\
\implies (2 \cdot \mathbb{P}[b_A = b^* | b_{tr} = 0] - 1) =& (2 \cdot \mathbb{P}[b_A = b^* | b_{tr} = 1] - 1) - \\
& 2 \cdot (2 \cdot \mathbb{P}[\text{Exp}^{\text{Ind-}tr}_{\mathscr{B}^4}(k) = 1] - 1) \\
\implies |2 \cdot \mathbb{P}[b_A = b^* | b_{tr} = 0] - 1| =& |(2 \cdot \mathbb{P}[b_A = b^* | b_{tr} = 1] - 1) - \\
& 2 \cdot (2 \cdot \mathbb{P}[\text{Exp}^{\text{Ind-}tr}_{\mathscr{B}^4}(k) = 1] - 1)| \\
\implies |2 \cdot \mathbb{P}[b_A = b^* | b_{tr} = 0] - 1| \leq& |2 \cdot \mathbb{P}[b_A = b^* | b_{tr} = 1] - 1| + \\
& 2 \cdot |2 \cdot \mathbb{P}[\text{Exp}^{\text{Ind-}tr}_{\mathscr{B}^4}(k) = 1] - 1| \\
\implies \text{Adv}_{\text{Exp}^5}(\mathscr{A}) \leq& \text{Adv}_{\text{Exp}^6}(\mathscr{A}) + 2 \cdot \text{Adv}^{\text{Ind-tr}}(\mathscr{B}^4)
\end{aligned}$$

Experimentos 6 y 7

La diferencia entre estos experimentos es que la simulación de la desencriptación se realiza de manera diferente. Se puede plantear un adversario IND-CPA, explicitado en el algoritmo 9.

Algoritmo 9 Adversario $\mathscr{B}^5$

function $\mathscr{B}^5(\omega)$
  $b^* \xleftarrow{\$} \{0,1\}$
  $b_A \xleftarrow{\$} \text{Exp}^{6,7}(b^*, \text{ENCRYPT}, \mathscr{A})$
  $b_B \leftarrow b_A = b^*$
  return b_B
end function
function ENCRYPT(M)
  $\pi^1 \xleftarrow{\$} S_t$ ▷ S_t es el conjunto de las permutaciones de t elementos
  $\pi^2 \xleftarrow{\$} S_t$
  for $i = 1...|M|$ **do**
    $C_i \leftarrow \text{LOR}(\pi^1(M)_i, \pi^2(M)_i)$
  end for
  return $\vec{C}$
end function
function DECRYPT($\vec{C}$)
  $\pi^A \leftarrow \text{COMPUTEPERMUTATION}()$
  $M' \leftarrow \pi^A \circ \pi^1(M_0)$
  Return M'
end function

En este adversario, $\text{Exp}^{6,7}$ consiste en llevar a cabo el Experimento 6 (ó 7) utilizando el bit b^* para escoger la permutación de los mensajes de los usuarios honestos.

La función ENCRYPT es utilizada por el servidor de mezcla honesto al momento de generar la salida, donde M_0 representa la lista inicial de mensajes ingresados (la lista M_b más los mensajes de usuarios corruptos). Es evidente que, desde el punto de vista de $\mathscr{A}$, el servidor de mezcla honesto se comporta de la misma manera que en los experimentos 6 y 7. Por otro lado, la función DECRYPT se utiliza al desencriptar los textos cifrados. La expresión π^A se utiliza para representar la composición de las permutaciones aplicadas por todos los servidores de mezcla corruptos, obtenida mediante el mecanismo de extracción de las demostraciones de correctitud (esto es, COMPUTEPERMUTATION).

Es fácil notar que si el oráculo LOR entregó textos cifrados correspondientes a su primer argumento (es decir, $b_{\text{ind-cpa}} = 0$), entonces la permutación de los mensajes emitida por el algoritmo DECRYPT es la correcta, con lo que $\mathscr{A}$ se enfrenta al Experimento 6. Por otro lado, si $b_{\text{ind-cpa}} = 1$, la permutación entregada es aleatoria, con lo que el adversario se enfrenta al Experimento 7.

Así,

$$\begin{aligned}
\mathbb{P}[\text{Exp}^{\text{IND-CPA}}_{\mathscr{B}^5}(k)=1] =& \mathbb{P}[b_B = b_{\text{ind-cpa}}] \\
=& \mathbb{P}[b_B = b_{\text{ind-cpa}} | b_{\text{ind-cpa}} = 1] \cdot \mathbb{P}[b_{\text{ind-cpa}} = 1] + \\
& \mathbb{P}[b_B = b_{\text{ind-cpa}} | b_{\text{ind-cpa}} = 0] \cdot \mathbb{P}[b_{\text{ind-cpa}} = 0] \\
=& \frac{1}{2}\mathbb{P}[b_A = b^* | b_{\text{ind-cpa}} = 1] + \frac{1}{2}\mathbb{P}[b_A \neq b^* | b_{\text{ind-cpa}} = 0] \\
=& \frac{1}{2}(1 + \mathbb{P}[b_A = b^* | b_{\text{ind-cpa}} = 1] - \mathbb{P}[b_A = b^* | b_{\text{ind-cpa}} = 0])
\end{aligned}$$

Luego:

$$\begin{aligned}
2 \cdot (2 \cdot \mathbb{P}[\text{Exp}^{\text{IND-CPA}}_{\mathscr{B}^5}(k)=1] - 1) =& (2 \cdot \mathbb{P}[b_A = b^* | b_{ind-cpa} = 1] - 1) - \\
& (2 \cdot \mathbb{P}[b_A = b^* | b_{ind-cpa} = 0] - 1) \\
\implies (2 \cdot \mathbb{P}[b_A = b^* | b_{ind-cpa} = 0] - 1) =& (2 \cdot \mathbb{P}[b_A = b^* | b_{ind-cpa} = 1] - 1) - \\
& 2 \cdot (2 \cdot \mathbb{P}[\text{Exp}^{\text{IND-CPA}}_{\mathscr{B}^5}(k) = 1] - 1) \\
\implies |2 \cdot \mathbb{P}[b_A = b^* | b_{ind-cpa} = 0] - 1| =& |(2 \cdot \mathbb{P}[b_A = b^* | b_{ind-cpa} = 1] - 1) - \\
& 2 \cdot (2 \cdot \mathbb{P}[\text{Exp}^{\text{IND-CPA}}_{\mathscr{B}^5}(k) = 1] - 1)| \\
\implies |2 \cdot \mathbb{P}[b_A = b^* | b_{ind-cpa} = 0] - 1| \leq& |2 \cdot \mathbb{P}[b_A = b^* | b_{ind-cpa} = 1] - 1| + \\
& 2 \cdot |2 \cdot \mathbb{P}[\text{Exp}^{\text{IND-CPA}}_{\mathscr{B}^5}(k) = 1] - 1| \\
\implies \text{Adv}_{\text{Exp}^6}(\mathscr{A}) \leq& \text{Adv}_{\text{Exp}^7}(\mathscr{A}) + 2 \cdot \text{Adv}^{\text{IND-CPA}}_{DLIN}(\mathscr{B}^5)
\end{aligned}$$

Experimentos 7 y 8

La única diferencia entre los experimentos 7 y 8 es que se utilizan mensajes distintos para los usuarios honestos. De esta forma, se plantea el adversario explicitado en el algoritmo 10 para el experimento IND-CPA. Sea $b_{ind-cpa}$ el bit aleatorio escogido en dicho experimento (según lo definido en el experimento 1, en la subsección 1.3.1).

Algoritmo 10 Adversario $\mathscr{B}^6$

```
function B^6(ω)
    b* ←$ {0,1}
    b_A ←$ Exp^{7,8}(b*, ENCRYPT, A)
    b_B ← b_A = b*
    return b_B
end function
function ENCRYPT(m)
    return LOR(m, 1)
end function
```

En el adversario $\mathscr{B}^6$, $\text{Exp}^{7,8}$ consiste en llevar a cabo el Experimento 7 (u 8) utilizando el bit

b^* para escoger la permutación de los mensajes de los usuarios honestos. Por otro lado, la función ENCRYPT es utilizada al momento de encriptar los mensajes de estos usuarios. Es claro que, en el caso de que el bit $b_{\text{ind-cpa}}$ del experimento IND-CPA sea 0, la interacción experimentada por $\mathscr{A}$ será idéntica a la del Experimento 7; si $b_{\text{ind-cpa}}$ es 1, estas interacciones serán idénticas a las del Experimento 8. De esta forma, por la definición del experimento IND-CPA, y condicionando sobre el bit b_B emitido por $\mathscr{B}^4$, se tiene:

$$\begin{aligned}
\mathbb{P}[\text{Exp}^{\text{IND-CPA}}_{\mathscr{B}^6}(k)=1] =& \mathbb{P}[b_B = b_{\text{ind-cpa}}] \\
=& \mathbb{P}[b_B = b_{\text{ind-cpa}} | b_{\text{ind-cpa}} = 1] \cdot \mathbb{P}[b_{\text{ind-cpa}} = 1] + \\
& \mathbb{P}[b_B = b_{\text{ind-cpa}} | b_{\text{ind-cpa}} = 0] \cdot \mathbb{P}[b_{\text{ind-cpa}} = 0] \\
=& \frac{1}{2}\mathbb{P}[b_A = b^* | b_{\text{ind-cpa}} = 1] + \frac{1}{2}\mathbb{P}[b_A \neq b^* | b_{\text{ind-cpa}} = 0] \\
=& \frac{1}{2}(1 + \mathbb{P}[b_A = b^* | b_{\text{ind-cpa}} = 1] - \mathbb{P}[b_A = b^* | b_{\text{ind-cpa}} = 0])
\end{aligned}$$

Luego:

$$\begin{aligned}
2 \cdot (2 \cdot \mathbb{P}[\text{Exp}^{\text{IND-CPA}}_{\mathscr{B}^6}(k)=1] - 1) =& (2 \cdot \mathbb{P}[b_A = b^* | b_{ind-cpa} = 1] - 1) - \\
& (2 \cdot \mathbb{P}[b_A = b^* | b_{ind-cpa} = 0] - 1) \\
\implies (2 \cdot \mathbb{P}[b_A = b^* | b_{ind-cpa} = 0] - 1) =& (2 \cdot \mathbb{P}[b_A = b^* | b_{ind-cpa} = 1] - 1) - \\
& 2 \cdot (2 \cdot \mathbb{P}[\text{Exp}^{\text{IND-CPA}}_{\mathscr{B}^6}(k)=1] - 1) \\
\implies |2 \cdot \mathbb{P}[b_A = b^* | b_{ind-cpa} = 0] - 1| =& |(2 \cdot \mathbb{P}[b_A = b^* | b_{ind-cpa} = 1] - 1) - \\
& 2 \cdot (2 \cdot \mathbb{P}[\text{Exp}^{\text{IND-CPA}}_{\mathscr{B}^6}(k)=1] - 1)| \\
\implies |2 \cdot \mathbb{P}[b_A = b^* | b_{ind-cpa} = 0] - 1| \leq& |2 \cdot \mathbb{P}[b_A = b^* | b_{ind-cpa} = 1] - 1| + \\
& 2 \cdot |2 \cdot \mathbb{P}[\text{Exp}^{\text{IND-CPA}}_{\mathscr{B}^6}(k)=1] - 1| \\
\implies \text{Adv}_{\text{Exp}^7}(\mathscr{A}) \leq& \text{Adv}_{\text{Exp}^8}(\mathscr{A}) + 2 \cdot \text{Adv}^{\text{IND-CPA}}_{DLIN}(\mathscr{B}^6)
\end{aligned}$$

Experimento 8

Finalmente, es útil notar que el bit b no es utilizado de ninguna forma en el Experimento Exp^8. Así, la probabilidad de que la salida del experimento sea 1 es exactamente $\frac{1}{2}$. Luego,

$$\text{Adv}_{\text{Exp}^8}(\mathscr{A}) = 0$$

Finalmente, pueden unirse estas ecuaciones para obtener:

$$\begin{aligned}
\text{Adv}^{\text{MIX-ANON}}_{\mathscr{MIX}}(\mathscr{A}) &= \text{Adv}_{\text{Exp}^1}(\mathscr{A}) \\
&\leq \text{Adv}_{\text{Exp}^2}(\mathscr{A}) + 2 \cdot \ \text{Adv}^{\text{Ind-}\omega}_{GS}(\mathscr{B}^1) \\
&\leq \text{Adv}_{\text{Exp}^3}(\mathscr{A}) + 2 \cdot \ \text{Adv}^{\text{Ind-}\omega}_{GS}(\mathscr{B}^1) + 2 \cdot \ \text{Adv}^{\text{Ind-}\omega}_{GS}(\mathscr{B}^2)
\end{aligned}$$

$$
\begin{aligned}
&= \mathrm{Adv}_{\mathrm{Exp}^4}(\mathscr{A}) + 2\cdot\ \mathrm{Adv}^{\text{Ind-}\omega}_{GS}(\mathscr{B}^1) + 2\cdot\ \mathrm{Adv}^{\text{Ind-}\omega}_{GS}(\mathscr{B}^2) \\
&\leq \mathrm{Adv}_{\mathrm{Exp}^5}(\mathscr{A}) + 2\cdot\ \mathrm{Adv}^{\text{Ind-}\omega}_{GS}(\mathscr{B}^1) + 2\cdot\ \mathrm{Adv}^{\text{Ind-}\omega}_{GS}(\mathscr{B}^2) + 2\cdot\ \mathrm{Adv}^{\text{Ind-zk}}(\mathscr{B}^3) \\
&\leq \mathrm{Adv}_{\mathrm{Exp}^6}(\mathscr{A}) + 2\cdot\ \mathrm{Adv}^{\text{Ind-}\omega}_{GS}(\mathscr{B}^1) + 2\cdot\ \mathrm{Adv}^{\text{Ind-}\omega}_{GS}(\mathscr{B}^2) + 2\cdot\ \mathrm{Adv}^{\text{Ind-zk}}(\mathscr{B}^3) + \\
&\quad 2\cdot\ \mathrm{Adv}^{\text{Ind-tr}}(\mathscr{B}^4) \\
&\leq \mathrm{Adv}_{\mathrm{Exp}^7}(\mathscr{A}) + 2\cdot\ \mathrm{Adv}^{\text{Ind-}\omega}_{GS}(\mathscr{B}^1) + 2\cdot\ \mathrm{Adv}^{\text{Ind-}\omega}_{GS}(\mathscr{B}^2) + 2\cdot\ \mathrm{Adv}^{\text{Ind-zk}}(\mathscr{B}^3) + \\
&\quad 2\cdot\ \mathrm{Adv}^{\text{Ind-tr}}(\mathscr{B}^4) + 2\cdot\ \mathrm{Adv}^{\text{IND-CPA}}_{DLIN}(\mathscr{B}^5) \\
&\leq \mathrm{Adv}_{\mathrm{Exp}^8}(\mathscr{A}) + 2\cdot\ \mathrm{Adv}^{\text{Ind-}\omega}_{GS}(\mathscr{B}^1) + 2\cdot\ \mathrm{Adv}^{\text{Ind-}\omega}_{GS}(\mathscr{B}^2) + 2\cdot\ \mathrm{Adv}^{\text{Ind-zk}}(\mathscr{B}^3) + \\
&\quad 2\cdot\ \mathrm{Adv}^{\text{Ind-tr}}(\mathscr{B}^4) + 2\cdot\ \mathrm{Adv}^{\text{IND-CPA}}_{DLIN}(\mathscr{B}^5) + 2\cdot\ \mathrm{Adv}^{\text{IND-CPA}}_{DLIN}(\mathscr{B}^6) \\
&= \mathrm{Adv}_{\text{Ind-}\omega}(\mathscr{B}^1) + 2\cdot\ \mathrm{Adv}^{\text{Ind-}\omega}(\mathscr{B}^2) + 2\cdot\ \mathrm{Adv}^{\text{Ind-zk}}(\mathscr{B}^3) + \\
&\quad 2\cdot\ \mathrm{Adv}^{\text{Ind-tr}}(\mathscr{B}^4) + 2\cdot\ \mathrm{Adv}^{\text{IND-CPA}}_{DLIN}(\mathscr{B}^5) + 2\cdot\ \mathrm{Adv}^{\text{IND-CPA}}_{DLIN}(\mathscr{B}^6)
\end{aligned}
$$

Luego se tiene que

$$
\begin{aligned}
\mathrm{Adv}^{\text{MIX-ANON}}_{\mathscr{MIX}}(\mathscr{A}) \leq\ & 2\cdot\ \mathrm{Adv}^{\text{Ind-}\omega}_{GS}(\mathscr{B}^1) + 2\cdot\ \mathrm{Adv}^{\text{Ind-}\omega}_{GS}(\mathscr{B}^2) + 2\cdot\ \mathrm{Adv}^{\text{Ind-zk}}(\mathscr{B}^3) + \\
& 2\cdot\ \mathrm{Adv}^{\text{Ind-tr}}(\mathscr{B}^4) + 2\cdot\ \mathrm{Adv}^{\text{IND-CPA}}_{DLIN}(\mathscr{B}^5) + 2\cdot\ \mathrm{Adv}^{\text{IND-CPA}}_{DLIN}(\mathscr{B}^6)
\end{aligned}
$$

Todos los términos del lado derecho de la inecuación son pequeños, debido a las propiedades de las herramientas utilizadas: indistinguibilidad de cadenas de referencia común, propiedad de Zero Knowledge especial con verificador honesto, emulación extendida con testigo e indistinguibilidad ante ataques de texto plano escogido. Con esto, se concluye la demostración. □

5.2 Infalsificabilidad del esquema

Se busca demostrar, a continuación, el siguiente lema:

Lema 2. *Para cualquier adversario de tiempo polinomial $\mathscr{A}$, la ventaja $\mathrm{Adv}^{\text{MIX-UF}}_{\mathscr{MIX}}(\mathscr{A})$ es pequeña.*

En efecto, para todo adversario $\mathscr{A}$ tal que $\mathrm{Adv}^{\text{MIX-UF}}_{\mathscr{MIX}}(\mathscr{A})$ sea no despreciable, puede construirse un adversario $\mathscr{B}$ también con ventaja no despreciable que rompe la propiedad de infalsificabilidad de firmas sobre textos cifrados del esquema, lo que entrega una cota para la ventaja de $\mathscr{A}$.

El adversario $\mathscr{B}$ se especifica a continuación. Dado que $\mathscr{B}$ es un adversario para la propiedad de infalsificabilidad de firmas sobre textos cifrados según lo expuesto en el Capítulo 3, recibe tanto las claves de encriptación como de desencriptación. De esta forma, escoge uno de estos pares de claves, de modo de utilizarlos como las claves de encriptación y desencriptación de la red de mezcla.

Así, $\mathscr{B}$ puede simular directa y perfectamente todos los protocolos de la red de mezcla, excepto por el firmado de los mensajes encriptados y la demostración de conocimiento de las claves de firmado.

La demostración de conocimiento puede ser simulada utilizando la metodología de Groth-Sahai. Por otro lado, para poder simular las n diferentes autoridades de certificación, $\mathscr{B}$ realiza los siguientes pasos:

- Luego de obtener $vk = Y$ como parte de su entrada, elige $i \xleftarrow{\$} \{1,...,k\}$.
- Se fija $Y_i \leftarrow Y$
- Para todo $j \in \{1,...,k\}$ tal que $j \neq i$, se genera $(Y_j, Z_j) \xleftarrow{\$} \text{KEYGEN}(\lambda)$.
- Se ejecuta el adversario $\mathscr{A}$. Para cada consulta al oráculo de firmas, $\mathscr{B}$ realiza lo siguiente:
  - Si la petición es para obtener una firma válida bajo la clave pública y_i, $\mathscr{B}$ utiliza su oráculo de firmas para obtener una firma válida y entregarla a $\mathscr{A}$.
  - Si la petición es para obtener una firma válida bajo la clave pública y_j con $j \neq i$, $\mathscr{B}$ utiliza la clave Z_j para elaborar una firma válida y entregarla a $\mathscr{A}$.
- El adversario $\mathscr{A}$ retorna (Y_j, C, σ).
- Si $Y_j \neq Y$, se retorna $\perp$. Si no, se retorna (C, σ) como una falsificación válida para la clave de verificación Y.

Es claro que si $\mathscr{B}$ no retorna $\perp$, entonces retorna una falsificación válida. Esto ocurre sólo si $Y_j = Y$, para el Y_j retornado por el adversario. Sin embargo, todos los Y_j distribuyen de la misma forma (ya que han sido generados utilizando el algoritmo KEYGEN). De esta forma, la probabilidad de que $Y_j = Y$ es $\frac{1}{k}$. Finalmente:

$$\text{Adv}^{\text{MIX-UF}}_{\mathscr{M}\mathscr{I}\mathscr{X}}(\mathscr{A}) \leq k \cdot \text{Adv}^{\text{c-uf}}_{\mathscr{M}\mathscr{I}\mathscr{X}}(\mathscr{B})$$

con $\text{Adv}^{\text{c-uf}}_{\mathscr{M}\mathscr{I}\mathscr{X}}(\mathscr{B})$ la ventaja de $\mathscr{B}$ en el experimento de infalsificabilidad de firmas sobre textos cifrados, definido en el Capítulo 3.

Queda demostrar que el esquema de encriptación específico utilizado en esta memoria es un esquema de firmas sobre mensajes encriptados realeatorizables infalsificable (lo que produce una cota para la ventaja de $\mathscr{B}$), y que es un esquema de encriptación con indistinguibilidad ante ataques de texto plano escogido. Dichas demostraciones se describen en el Apéndice.

Capítulo 6

Conclusiones

En este trabajo se llevó a cabo la construcción de una una red de comunicación anónima basada en reencriptación que permitiera la verificación de alguna propiedad criptográfica de manera pública. Se realizó el análisis de diversas definiciones y esquemas relacionados, con el objetivo de encontrar herramientas que sirvieran de componentes para una posible solución.

Como resultado de este proceso, se logró el desarrollo de un esquema que resuelve la problemática planteada, en base a la composición de diferentes herramientas criptográficas como encriptación homomórfica, sistemas de demostración y esquemas de firmado. Se demostró, utilizando técnicas estándares, la seguridad de la solución propuesta, validándola como tal.

La solución construida, sin embargo, no es particularmente eficiente. El tamaño de los textos cifrados generados crece linealmente con el número de bits, lo que limita su utilidad a aplicaciones en las que el tamaño de los mensajes se encuentre relativamente acotado. Adicionalmente, problemáticas como la desencriptación de los mensajes de manera distribuida se dejaron fuera del ámbito de este trabajo. Se detallan a continuación estas limitaciones, sus razones y las posibles mejoras para el esquema desarrollado.

6.1 Posibles mejoras y trabajo futuro

Existen diversas formas de mejorar el esquema propuesto en este trabajo. Se listan a continuación algunas de éstas:

6.1.1 Instanciación de políticas de filtrado más complejas

Como se especificó en el Capítulo 4, puede expandirse el proceso de firma de mensajes, requiriendo la autoridad certificadora que el usuario demuestre una propiedad sobre su mensaje encriptado. Esto permitiría un filtrado con una semántica más compleja (como censura de palabras específicas o el gasto de una moneda electrónica). Esquemas como los expuestos en [39] indican que este camino puede ser, de hecho, factible.

No obstante, se requiere que esta demostración sea públicamente verificable. De lo contrario, una autoridad de certificación de mensajes podría, arbitrariamente, negar una firma sobre un texto con las propiedades correctas. Esto, eventualmente, violaría el anonimato del sistema. Por ejemplo, en el caso de que se niegue el firmado de mensajes para un usuario específico, se revelarían los mensajes que éste quería enviar a la red (ya que serían los únicos mensajes no emitidos por ella).

6.1.2 Construcción de un esquema de firmas más eficiente

Uno de los factores que afectan la eficiencia del esquema propuesto es el hecho de que el esquema de firmas subyacente realiza el firmado sobre la representación binaria del mensaje. Esto resulta en la necesidad de realizar un commitment para cada bit del mensaje, resultando en textos cifrados que crecen (en cantidad de elementos de grupo) de forma lineal con el largo en bits del mensaje. De esta forma, un esquema de firmas que firme mensajes representados utilizando una cantidad constante de elementos de grupo reduciría en gran medida el tamaño de los textos cifrados del sistema.

Sin embargo, esta mejora se aprecia como dificultosa. En general, los esquemas de firma existentes involucran funciones de hash que impiden la codificación de una firma como ecuaciones demostrables mediante la metodología de Groth-Sahai. Así, si bien el sistema de demostración de Groth-Sahai posee una potencia considerable, su compatibilidad con esquemas de firmas es limitada.

6.1.3 Uso de desencriptación distribuida

En el esquema propuesto se asume que la desencriptación de los mensajes es llevada a cabo por una autoridad confiable. Sin embargo, en una implementación real este proceso debe llevarse a cabo por un número de servidores de desencriptación, con el objetivo de asegurar la recuperación correcta de los mensajes emitidos por los usuarios.

Esta extensión requiere adaptar el esquema de encriptación para proveerle las características de un esquema de encriptación umbral. Existen distintos esfuerzos por extender el esquema de

encriptación lineal utilizado en esta dirección, como se expone en [2], [36], y [21], lo que apunta a que es un problema factible de resolver. Sin embargo, puede requerir cambios en otros puntos del esquema, ya se debe ser cuidadoso de preservar la realeatorizabilidad de las firmas al introducir cambios en el esquema de encriptación.

6.1.4 Utilización de propiedades de encriptación más sofisticadas

Si bien se utilizó la propiedad de indistinguibilidad ante ataques de texto plano conocido para la demostración de seguridad del esquema, es razonable considerar que, al tener el texto cifrado una demostración de conocimiento del texto plano subyacente, el esquema posea una propiedad de seguridad más fuerte. Así, una posible extensión de este trabajo pudiera consistir en demostrar que el esquema posee la propiedad RCCA, expuesta en [44], estrictamente más fuerte que lo demostrado.

Por otro lado, demostrar propiedades más fuertes que las utilizadas sobre componentes del esquema podría ser útil para simplificarlo, al permitir la eliminación de algunas componentes (como alguna de las demostraciones de conocimiento involucradas).

6.1.5 Demostración de seguridad en el Modelo de Composición Universal

Por simplicidad, se realizó la demostración de la seguridad del sistema en el modelo de simulaciones. Sin embargo, es razonable considerar que pudiera construirse una demostración de seguridad en el modelo de Composición Universal. Esto permitiría asegurar la seguridad del sistema incluso cuando existieran múltiples instancias del protocolo ejecutándose de manera simultánea.

Las dificultades presentes para lograr esto son, principalmente, dos. Por un lado, debe formularse una funcionalidad ideal que modele apropiadamente el comportamiento del sistema. Este proceso es no trivial, ya que es necesario conjugar apropiadamente el anonimato y la infalsificabilidad del sistema. La segunda dificultad es realizar la demostración en sí de que el esquema concreta la funcionalidad ideal formulada.

Referencias

[1] An, J., Y. Dodis y T. Rabin: *On the security of joint signature and encryption*. En Knudsen, L. (ed.): *Advances in Cryptology — EUROCRYPT 2002*, vol. 2332 de *Lecture Notes in Computer Science*, págs. 83–107. Springer Berlin Heidelberg, 2002, ISBN 978-3-540-43553-2. http://dx.doi.org/10.1007/3-540-46035-7_6.

[2] Arita, S. y K. Tsurudome: *Construction of threshold public-key encryptions through tag-based encryptions*. En Abdalla, M., D. Pointcheval, P. A. Fouque y D. Vergnaud (eds.): *Applied Cryptography and Network Security*, vol. 5536 de *Lecture Notes in Computer Science*, págs. 186–200. Springer Berlin Heidelberg, 2009, ISBN 978-3-642-01956-2. http://dx.doi.org/10.1007/978-3-642-01957-9_12.

[3] Arora, S. y B. Barak: *Computational Complexity: A Modern Approach*. Cambridge University Press, New York, NY, USA, 1st ed., 2009, ISBN 0521424267, 9780521424264.

[4] Baek, J., R. Safavi-Naini y W. Susilo: *Public key encryption with keyword search revisited*. En Gervasi, O., B. Murgante, A. Laganà, D. Taniar, Y. Mun y M. Gavrilova (eds.): *Computational Science and Its Applications – ICCSA 2008*, vol. 5072 de *Lecture Notes in Computer Science*, págs. 1249–1259. Springer Berlin Heidelberg, 2008, ISBN 978-3-540-69838-8. http://dx.doi.org/10.1007/978-3-540-69839-5_96.

[5] Barratt, M. J.: *Silk road: Ebay for drugs*. Addiction, 107(3):683–683, 2012, ISSN 1360-0443. http://dx.doi.org/10.1111/j.1360-0443.2011.03709.x.

[6] Bellare, M. y P. Rogaway: *The security of triple encryption and a framework for code-based game-playing proofs*. En Vaudenay, S. (ed.): *Advances in Cryptology - EUROCRYPT 2006*, vol. 4004 de *Lecture Notes in Computer Science*, págs. 409–426. Springer Berlin Heidelberg, 2006, ISBN 978-3-540-34546-6. http://dx.doi.org/10.1007/11761679_25.

[7] Blanchet, B. y D. Pointcheval: *Automated security proofs with sequences of games*. En *Proc. 27th IEEE Symposium on Security*, págs. 537–554. Springer, 2006.

[8] Blazy, O., S. Canard, G. Fuchsbauer, A. Gouget, H. Sibert y J. Traoré: *Achieving optimal anonymity in transferable e-cash with a judge*. En *Proceedings of the 4th international conference on Progress in cryptology in Africa*, AFRICACRYPT'11, págs. 206–223, Berlin, Heidelberg, 2011. Springer-Verlag, ISBN 978-3-642-21968-9. http://dl.acm.org/citation.cfm?id=2026469.2026487.

[9] Blazy, O., G. Fuchsbauer, D. Pointcheval y D. Vergnaud: *Signatures on randomizable ciphertexts*. En *Proceedings of the 14th international conference on Practice and theory in public key cryptography conference on Public key cryptography*, PKC'11, págs. 403–422, Berlin, Heidelberg, 2011. Springer-Verlag, ISBN 978-3-642-19378-1. http://dl.acm.org/citation.cfm?id=1964658.1964692.

[10] Blum, M.: *Coin flipping by telephone a protocol for solving impossible problems*. SIGACT News, 15(1):23–27, Ene. 1983, ISSN 0163-5700. http://doi.acm.org/10.1145/1008908.1008911.

[11] Blum, M., P. Feldman y S. Micali: *Non-interactive zero-knowledge and its applications*. En *Proceedings of the twentieth annual ACM symposium on Theory of computing*, STOC '88, págs. 103–112, New York, NY, USA, 1988. ACM, ISBN 0-89791-264-0. http://doi.acm.org/10.1145/62212.62222.

[12] Boneh, D.: *The decision diffie-hellman problem*. En Buhler, J. (ed.): *Algorithmic Number Theory*, vol. 1423 de *Lecture Notes in Computer Science*, págs. 48–63. Springer Berlin Heidelberg, 1998, ISBN 978-3-540-64657-0. http://dx.doi.org/10.1007/BFb0054851.

[13] Boneh, D., X. Boyen y S. Halevi: *Chosen ciphertext secure public key threshold encryption without random oracles*. En *RSA-CT*, págs. 226–243, 2006. http://www.truststc.org/pubs/603.html.

[14] Boneh, D., X. Boyen y H. Shacham: *Short group signatures*. En *Advances in Cryptology—CRYPTO 2004*, vol. 3152 de *Lecture Notes in Computer Science*, págs. 41–55. Berlin: Springer-Verlag, 2004. Disponible en http://www.cs.stanford.edu/~xb/crypto04a/.

[15] Boneh, D., C. Gentry, B. Lynn y H. Shacham: *Aggregate and verifiably encrypted signatures from bilinear maps*. En *Proceedings of the 22nd international conference on Theory and applications of cryptographic techniques*, EUROCRYPT'03, págs. 416–432, Berlin, Heidelberg, 2003. Springer-Verlag, ISBN 3-540-14039-5. http://dl.acm.org/citation.cfm?id=1766171.1766207.

[16] Camenisch, J., S. Hohenberger y A. Lysyanskaya: *Compact e-cash*. En *In EUROCRYPT, volume 3494 of LNCS*, págs. 302–321. Springer-Verlag, 2005.

[17] Camenisch, J., A. Lysyanskaya y M. Meyerovich: *Endorsed e-cash*. En *Proceedings of the 2007 IEEE Symposium on Security and Privacy*, SP '07, págs. 101–115, Washington, DC, USA, 2007. IEEE Computer Society, ISBN 0-7695-2848-1. http://dx.doi.org/10.1109/SP.2007.15.

[18] Camenisch, J. y A. Mityagin: *Mix-network with stronger security*. En *Proceedings of the 5th international conference on Privacy Enhancing Technologies*, PET'05, págs. 128–146, Berlin, Heidelberg, 2006. Springer-Verlag, ISBN 3-540-34745-3, 978-3-540-34745-3. http://dx.doi.org/10.1007/11767831_9.

[19] Canetti, R.: *Universally composable security: A new paradigm for cryptographic protocols*. Cryptology ePrint Archive, Report 2000/067, 2000. http://eprint.iacr.org/.

[20] Chase, M., M. Kohlweiss, A. Lysyanskaya y S. Meiklejohn: *Malleable proof systems and applications*. Cryptology ePrint Archive, Reporte 2012/012, 2012. http://eprint.iacr.org/.

[21] Chase, M., M. Kohlweiss, A. Lysyanskaya y S. Meiklejohn: *Verifiable elections that scale for free*. En Kurosawa, K. y G. Hanaoka (eds.): *Public-Key Cryptography – PKC 2013*, vol. 7778 de *Lecture Notes in Computer Science*, págs. 479–496. Springer Berlin Heidelberg, 2013, ISBN 978-3-642-36361-0. http://dx.doi.org/10.1007/978-3-642-36362-7_29.

[22] Chaum, D.: *Blind signatures for untraceable payments*. En *Advances in Cryptology: Proceedings of CRYPTO '82*, págs. 199–203. Plenum, 1982.

[23] Chaum, D. y T. P. Pedersen: *Transferred cash grows in size*. En *Proceedings of the 11th annual international conference on Theory and application of cryptographic techniques*, EUROCRYPT'92, págs. 390–407, Berlin, Heidelberg, 1993. Springer-Verlag, ISBN 3-540-56413-6. http://dl.acm.org/citation.cfm?id=1754948.1754992.

[24] Chaum, D. L.: *Untraceable electronic mail, return addresses, and digital pseudonyms*. Commun. ACM, 24(2):84–90, Feb. 1981, ISSN 0001-0782. http://doi.acm.org/10.1145/358549.358563.

[25] Christopherson, K. M.: *The positive and negative implications of anonymity in internet social interactions: "On the Internet, Nobody Knows You're a Dog"*. Computers in Human Behavior, 23(6):3038 – 3056, 2007, ISSN 0747-5632. http://www.sciencedirect.com/science/article/pii/S0747563206001221.

[26] Crescenzo, G. D., J. Katz, R. Ostrovsky y A. Smith: *Efficient and non-interactive non-malleable commitment*. Cryptology ePrint Archive, Report 2001/032, 2001. http://eprint.iacr.org/.

[27] Davenport, D.: *Anonymity on the internet: why the price may be too high*. Commun. ACM, 45(4):33–35, Abr. 2002, ISSN 0001-0782. http://doi.acm.org/10.1145/505248.505267.

[28] Delfs, H. y H. Knebl: *Introduction to Cryptography: Principles and Applications*. Springer, 2002, ISBN 3-540-42278-1.

[29] Dent, A. W. y Y. Zheng: *Practical Signcryption*. Springer-Verlag New York, Inc., New York, NY, USA, 1st ed., 2010, ISBN 3540894098, 9783540894094.

[30] Federal Bureau of Investigation: *Complaint/assessment form FD-71*, 2011. http://s3.documentcloud.org/documents/367226/responsive-documents.pdf.

[31] Fuchsbauer, G.: *Commuting signatures and verifiable encryption*. En *Proceedings of the 30th Annual international conference on Theory and applications of cryptographic techniques: advances in cryptology*, EUROCRYPT'11, págs. 224–245, Berlin, Heidelberg, 2011. Springer-Verlag, ISBN 978-3-642-20464-7. http://dl.acm.org/citation.cfm?id=2008684.2008703.

[32] Fuchsbauer, G., D. Pointcheval y D. Vergnaud: *Transferable constant-size fair e-cash*. En Garay, J., A. Miyaji y A. Otsuka (eds.): *Cryptology and Network Security*, vol. 5888 de *Lecture Notes in Computer Science*, págs. 226–247. Springer Berlin Heidelberg, 2009, ISBN 978-3-642-10432-9. http://dx.doi.org/10.1007/978-3-642-10433-6_15.

[33] Galbraith, S., K. Paterson y N. Smart: *Pairings for cryptographers*. Cryptology ePrint Archive, Reporte 2006/165, 2006. http://eprint.iacr.org/.

[34] Goldreich, O., S. Micali y A. Wigderson: *Proofs that yield nothing but their validity or all languages in np have zero-knowledge proof systems*. J. ACM, 38(3):690–728, Jul 1991, ISSN 0004-5411. http://doi.acm.org/10.1145/116825.116852.

[35] Goldwasser, S. y S. Micali: *Probabilistic encryption: How to play mental poker keeping secret all partial information*. En *Proceedings of the Fourteenth Annual ACM Symposium on Theory of Computing*, STOC '82, págs. 365–377, New York, NY, USA, 1982. ACM, ISBN 0-89791-070-2. http://doi.acm.org/10.1145/800070.802212.

[36] Gordon, S., T. Malkin, M. Rosulek y H. Wee: *Multi-party computation of polynomials and branching programs without simultaneous interaction*. En Johansson, T. y P. Nguyen (eds.): *Advances in Cryptology – EUROCRYPT 2013*, vol. 7881 de *Lecture Notes in Computer Science*, págs. 575–591. Springer Berlin Heidelberg, 2013, ISBN 978-3-642-38347-2. http://dx.doi.org/10.1007/978-3-642-38348-9_34.

[37] Groth, J.: *A verifiable secret shuffle of homomorphic encryptions*. J. Cryptol., 23(4):546–579, Oct. 2010, ISSN 0933-2790. http://dx.doi.org/10.1007/s00145-010-9067-9.

[38] Groth, J. y A. Sahai: *Efficient non-interactive proof systems for bilinear groups*. En *Proceedings of the theory and applications of cryptographic techniques 27th annual international conference on Advances in cryptology*, EUROCRYPT'08, págs. 415–432, Berlin, Heidelberg, 2008. Springer-Verlag, ISBN 3-540-78966-9, 978-3-540-78966-6. http://dl.acm.org/citation.cfm?id=1788414.1788438.

[39] Ibraimi, L., S. Nikova, P. Hartel y W. Jonker: *Public-key encryption with delegated search*. En *Proceedings of the 9th international conference on Applied cryptography and network security*, ACNS'11, págs. 532–549, Berlin, Heidelberg, 2011. Springer-Verlag, ISBN 978-3-642-21553-7. http://dl.acm.org/citation.cfm?id=2025968.2026010.

[40] Katz, J. y Y. Lindell: *Introduction to Modern Cryptography (Chapman & Hall/Crc Cryptography and Network Security Series)*. Chapman & Hall/CRC, 2007, ISBN 1584885513.

[41] Meiklejohn, S.: *An extension of the Groth-Sahai proof system*. Tesis de Magister en Universidad Brown, 2009. http://cseweb.ucsd.edu/~smeiklejohn/files/mastersthesis.pdf.

[42] Naehrig, M.: *Pairings for cryptography*. Charla invitada en el seminario del Digital Security Group, Radboud Universiteit Nijmegen, Países Bajos, 2009.

[43] Park, C., K. Itoh y K. Kurosawa: *Efficient anonymous channel and all/nothing election scheme*. En Helleseth, T. (ed.): *Advances in Cryptology — EUROCRYPT '93*, vol. 765 de *Lecture Notes in Computer Science*, págs. 248–259. Springer Berlin Heidelberg, 1994, ISBN 978-3-540-57600-6. http://dx.doi.org/10.1007/3-540-48285-7_21.

[44] Prabhakaran, M. y M. Rosulek: *Rerandomizable rcca encryption*. Cryptology ePrint Archive, Report 2007/119, 2007. http://eprint.iacr.org/.

[45] Shoup, V.: *Sequences of games: a tool for taming complexity in security proofs*. Cryptology ePrint Archive, Report 2004/332, 2004. http://eprint.iacr.org/.

[46] Shoup, V.: *A Computational Introduction to Number Theory and Algebra*. Cambridge University Press, 2005. http://dx.doi.org/10.1017/CBO9781139165464.

[47] Wang, Y., M. Manulis, M. Au y W. Susilo: *Relations among privacy notions for signcryption and key invisible "sign-then-encrypt"*. En Boyd, C. y L. Simpson (eds.): *Information Security and Privacy*, vol. 7959 de *Lecture Notes in Computer Science*, págs. 187–202. Springer Berlin Heidelberg, 2013, ISBN 978-3-642-39058-6. http://dx.doi.org/10.1007/978-3-642-39059-3_13.

[48] Waters, B.: *Efficient identity-based encryption without random oracles*. En *Proceedings of the 24th annual international conference on Theory and Applications of Cryptographic Techniques*, EUROCRYPT'05, págs. 114–127, Berlin, Heidelberg, 2005. Springer-Verlag, ISBN 3-540-25910-4, 978-3-540-25910-7. http://dx.doi.org/10.1007/11426639_7.

[49] Zetter, K.: *Feds Arrest Alleged 'Dread Pirate Roberts', the Brain behind the Silk Road Drug Site*, 2013. http://www.wired.com/threatlevel/2013/10/silk-road-raided/.

[50] Zheng, Y.: *Digital signcryption or how to achieve cost(signature & encryption) << cost(signature) + cost(encryption)*. En *Proceedings of the 17th Annual International Cryptology Conference on Advances in Cryptology*, CRYPTO '97, págs. 165–179, London, UK, UK, 1997. Springer-Verlag, ISBN 3-540-63384-7. http://dl.acm.org/citation.cfm?id=646762.706181.

Apéndices

A Algoritmos del protocolo

A .1 Esquema de encriptación subyacente

Se presenta en esta sección, en las figuras 11, 12 y 13, el esquema de encriptación utilizado en la solución propuesta.

En SETUP, $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ es un *pairing* bilineal no degenerado; $\mathbb{G}$ y $\mathbb{G}_T$ son grupos de orden primo p, con generadores g y $e(g,g)$, respectivamente. Los algoritmos PROVER y PROVEM son una aplicación directa de demostraciones de Groth-Sahai:

- PROVER contiene commitments $\langle Y^{r_1} \rangle$, $\langle Y^{r_2} \rangle$; se incluyen además demostraciones para las siguientes ecuaciones, para $i \in \{1,2\}$:

$$e(\langle Y^{r_i} \rangle, X_i) = e(H_i, Y)$$

- PROVEM contiene commitments $\langle M_i \rangle$ para cada bit M_i del mensaje M, y commitments $\langle Y^{r_{1,i}} \rangle$, $\langle Y^{r_{2,i}} \rangle$ para cada $i \in \{1...|M|\}$; con esto, demuestra las siguientes ecuaciones:

$$e(H_3, Y) = e(u_0 \prod_{i=1}^{|M|} u_i^{\langle M_i \rangle}, Y) \cdot e(\langle Y^{r_1} \rangle \langle Y^{r_2} \rangle, g)$$
$$e(c_{i,1}, Y) = e(\langle Y^{r_{1,i}} \rangle, X_1)$$
$$e(c_{i,2}, Y) = e(\langle Y^{r_{2,i}} \rangle, X_2)$$

además de demostraciones de que cada $\langle M_i \rangle$ contiene un bit (es decir, que el valor contenido sólo puede ser 0 o 1).

Esquema 11 Algoritmos del esquema de encriptación

function SETUP(1^k)
$(p, \mathbb{G}, \mathbb{G}_T, e, g) \xleftarrow{\$}$ GENGROUP(1^k) ▷ Definición de los grupos bilineales
$h \xleftarrow{\$} \mathbb{G}$
$u_0, ..., u_l \xleftarrow{\$} \mathbb{G}$ ▷ l es el largo de los mensajes a utilizar
$u \leftarrow (u_0, ..., u_l)$
return $(p, \mathbb{G}, \mathbb{G}_T, e, g, h, u)$
end function

function EKEYGEN(1^k)
$x_1, x_2 \xleftarrow{\$} \mathbb{Z}_p$
$dk \leftarrow (x_1, x_2)$
$pk \leftarrow (X_1, X_2) = (g^{x_1}, g^{x_2})$
return (pk, dk)
end function

function ENCRYPT$_{pk=(X_1,X_2),vk=Y}(M)$
$r_1, r_2 \xleftarrow{\$} \mathbb{Z}_p$
$H \leftarrow (H_1, H_2, H_3) = (X_1^{r_1}, X_2^{r_2}, g^{r_1+r_2} \cdot \mathrm{F}(M))$
$r_{1,1}, ..., r_{1,k=|M|} \xleftarrow{\$} \mathbb{Z}_p$
$r_{2,1}, ..., r_{2,k=|M|} \xleftarrow{\$} \mathbb{Z}_p$
$c \leftarrow \{(X_1^{r_{1,i}}, X_2^{r_{2,i}}, g^{r_{1,i}+r_{2,i}} \cdot u_i^{m_i})\}_{i=1}^{k=|M|}$
$\Pi_r \leftarrow$ PROVER(Y, H, r_1, r_2)
$\Pi_M \leftarrow$ PROVEM$(Y, M, c, r_1, r_2, \{r_{1,i}\}, \{r_{2,i}\})$
return $C = (c, H, \Pi_r, \Pi_M)$
end function

function DECRYPT$_{dk=(x_1,x_2)}(C = (c, H, \Pi_r, \Pi_M), \sigma))$
if (VERIFYPROOFR$(H, \Pi_r) \neq true$ **or** VERIFYPROOFM$(H, \Pi_M)) \neq true$ **or** EVERIFY$_{vk}(C, \sigma) \neq true$ **then**
return $\perp$
end if
for $i = 1, ..., L$ **do**
$m_i \leftarrow DLog_{u_i}(c_{i,3} \cdot (c_{i,1}^{1/x_1} c_{i,2}^{1/x_2})^{-1})$ ▷ Esto es posible porque $m_i \in \{0, 1\}$
end for
return $M = m_1, ..., m_k$
end function

Esquema 12 Algoritmos del esquema de encriptación (continuación)

function SKEYGEN(1^k)
  $y \xleftarrow{\$} \mathbb{Z}_p$
  $vk \leftarrow Y = g^y$
  $sk \leftarrow Z = h^y$
  return (vk, sk)
end function

function ESIGN$_{sk=Z,pk=(X_1,X_2)}(C = (c, H, \Pi_r, \Pi_M))$
  if (VERIFYPROOFR(H, Π_r) **and** VERIFYPROOFM$(H, \Pi_M)) \neq true$ **then**
    return $\perp$
  end if
  $s \xleftarrow{\$} \mathbb{Z}_p$
  return $\sigma = (H_1^s, H_2^s, Z \cdot H_3^s, X_1^s, X_2^s, g^s)$
end function

function EVERIFY$_{vk=Y,pk=(X_1,X_2)}(C = (c, H, \Pi_r, \Pi_M), \sigma = (\sigma_1, \sigma_2, \sigma_3, \sigma_4, \sigma_5, \sigma_6))$
  if VERIFYPROOFS$(c, H, \Pi_r, \Pi_M) \neq true$ **then**
    return $false$
  end if
  if $e(\sigma_1, X_1) \neq e(H_1, \sigma_4)$ **or** $e(\sigma_1, g) \neq e(H_1, \sigma_6)$ **then**
    return $false$
  end if
  if $e(\sigma_2, X_2) \neq e(H_2, \sigma_5)$ **or** $e(\sigma_2, g) \neq e(H_2, \sigma_6)$ **then**
    return $false$
  end if
  return $e(\sigma_3, g) = e(h, Y) \cdot e(H_3, \sigma_6)$
end function

Esquema 13 Algoritmos del esquema de encriptación (continuación)

function $\text{RANDOM}_{vk=Y,pk=(X_1,X_2)}(C=(c,H,\Pi_r,\Pi_M),\sigma=(\sigma_1,\sigma_2,\sigma_3,\sigma_4,\sigma_5,\sigma_6))$

$r'_1, r'_2 \xleftarrow{\$} \mathbb{Z}_p$

$H' \leftarrow (H_1 \cdot X_1^{r'_1}, H_2 \cdot X_2^{r'_2}, H_3 \cdot g^{(}r'_1 + r'_2))$

$s' \xleftarrow{\$} \mathbb{Z}_p$

$\sigma' \leftarrow (\sigma_1 \cdot H_1^{s'} \cdot \sigma_4^{r'_1} \cdot X_1^{r'_1 s'}, \sigma_2 \cdot H_2^{s'} \cdot \sigma_5^{r'_2} \cdot X_2^{r'_2 s'}, \sigma_3 \cdot H_3^{s'} \cdot \sigma_6^{r'_1+r'_2} \cdot g^{(r'_1+r'_2)s'}, \sigma_4 \cdot X_1^{s'}, \sigma_5 \cdot X_2^{s'}, \sigma_6 \cdot g^{s'})$

$r'_{1,1}, ..., r'_{1,|M|} \xleftarrow{\$} \mathbb{Z}_p$

$r'_{2,1}, ..., r'_{2,|M|} \xleftarrow{\$} \mathbb{Z}_p$

$c'_i \leftarrow (c'_{i,1} \cdot X_1^{r'_{1,i}}, c'_{i,2} \cdot X_2^{r'_{2,i}}, c'_{i,3} \cdot g^{r'_{1,i}+r'_{2,i}}) \; \forall i = 1, ..., |M|$

$c' \leftarrow (c'_1, ..., c_{|M|})$

$\Pi'_r, \Pi'_M, \xleftarrow{\$} \text{RERANDPROOFS}(\Pi_r, \Pi_M, C, r'_1, r'_2, s', \{r'_{1,i}\}, \{r'_{2,i}\})$

$C' \leftarrow (c', H', \Pi'_r, \Pi'_M)$

return (C', σ')

end function

function F(M)

$m_1, ..., m_l \leftarrow M$

return $u_0 \cdot \prod_{i=1}^{|M|} u_i^{m_i}$

end function

A .2 Demostraciones utilizadas

De la misma forma que en [18], el esquema propuesto requiere de las demostraciones para funcionar:

Demostración del conocimiento de las claves de firmado: Esta demostración se realiza en la inicialización del esquema. Involucra el firmado de un mensaje escogido por la autoridad confiable $\mathscr{K}$. La metodología de Groth-Sahai permite demostrar que un commitment en particular contiene la clave que hace que la ecuación de correctitud se cumpla. El objetivo de esta demostración es permitir que las claves de firmado de cada autoridad sean extraídas en las simulaciones utilizadas para las demostraciones de seguridad del esquema.

Demostración de mezclado correcto: En este punto, demostrar el mezclado sólo de las componentes H de los textos cifrados es suficiente.

La probabilidad de falsificación de las firmas adjuntas a los mensajes es despreciable, lo que obliga a mezclar correctamente las firmas adjuntas; por otro lado, la consistencia del sistema de demostración obliga a mezclar correctamente las demás componentes del texto cifrado (c y Π). De lo contrario, la verificación de las firmas y/o las demostraciones adjuntas al texto cifrado fallarían.

La demostración del mezclado correcto de las componentes H de los textos cifrados puede realizarse siguiendo la metodología descrita en el trabajo de Groth [37]. Si bien H no es homomórfico, el mecanismo de realeatorización funciona de manera análoga a la multiplicación por la componente H de un texto cifrado en el que $M = 0^L$. De esta forma, la metodología de Groth permite demostrar el mezclado correcto de forma eficiente, tomando 7 rondas de comunicación entre las partes, y una cantidad de exponenciaciones en $\mathbb{G}$ lineal en t, la cantidad de mensajes a mezclar.

Demostración de igualdad de mensajes para dos textos cifrados: Análogamente al punto anterior, sólo es necesario demostrar la igualdad entre los contenidos de las componentes H y H' del antiguo y nuevo texto cifrado, respectivamente. Esto se traduce en mostrar que existen valores r_1, r_2, r_1', r_2' tal que:

$$H_1 = X_1^{r_1} \wedge H_2 = X_2^{r_2} \wedge H_1' = X_1^{r_1'} \wedge H_2' = X_2^{r_2'} \wedge H_3 \cdot H_3'^{-1} = g^{r_1+r_2-r_1'-r_2'}$$

ecuaciones que pueden ser demostradas con la metodología de Groth-Sahai.

B Demostraciones auxiliares

B .1 Seguridad semántica del esquema de encriptación subyacente

La seguridad semántica del esquema de encriptación utilizado en la solución propuesta queda demostrada por el siguiente lema:

Lema 3. *El esquema propuesto es un esquema de encriptación que posee indistinguibilidad de textos encriptados ante ataques de texto plano escogidos.*

Demostración. Para comenzar la demostración de este teorema, es útil definir el siguiente esquema $\mathscr{BLE}$ (*bitwise linear encryption*, encriptación lineal bit a bit).

Se busca acotar la ventaja IND-CPA de $\mathscr{A}$ para el esquema propuesto. Para comenzar, es útil considerar el adversario $\mathscr{B}$, explicitado en el algoritmo 15, para $\mathscr{BLE}$.

Es necesario calcular la ventaja de este adversario. De la definición de ventaja IND-CPA:

$$\mathrm{Adv}^{\text{IND-CPA}}(\mathscr{B}^1) = |2 \cdot \mathbb{P}[\mathrm{PubK}^{\text{eav}}_{\mathscr{B}^1,\mathscr{BLE}}(k) = 1] - 1|$$

Esta probabilidad es equivalente a $\mathbb{P}[\mathscr{B}^2(1^k) = 1]$, con el algoritmo $\mathscr{B}^2$ definido como el algoritmo 16). Esto se tiene de manera natural, ya que $\mathscr{B}^2$ reproduce perfectamente el experimento de confidencialidad.

Esquema 14 Esquema auxiliar $\mathscr{BLE}$

function KEYGEN(1^k)
  $x_1, x_2 \xleftarrow{\$} \mathbb{Z}_p$
  $dk \leftarrow (x_1, x_2)$
  $pk \leftarrow (X_1, X_2) = (g^{x_1}, g^{x_2})$
  return (pk, dk)
end function

function ENCRYPT$_{pk=(X_1,X_2)}(M)$
  $r_{1,1}, ..., r_{1,k=|M|} \xleftarrow{\$} \mathbb{Z}_p$
  $r_{2,1}, ..., r_{2,k=|M|} \xleftarrow{\$} \mathbb{Z}_p$
  $c \leftarrow \{(X_1^{r_{1,i}}, X_2^{r_{2,i}}, g^{r_{1,i}+r_{2,i}} \cdot u_i^{m_i})\}_{i=1}^{k=|M|}$
  return c
end function

function DECRYPT$_{dk=(x_1,x_2)}(c)$
  $m_i \leftarrow DLog_{u_i}(c_{i,3} \cdot (c_{i,1}^{1/x_1} \cdot c_{i,2}^{1/x_2})^{-1})$ ▷ Esto es posible porque $m_i \in \{0,1\}$
  return $M = m_1, ..., m_k$
end function

Algoritmo 15 Adversario $\mathscr{B}^1$ para $\mathscr{BLE}$

function $\mathscr{B}_1^1(1^k)$
  $(\omega_{sim}, \tau) \xleftarrow{\$} S_1(1^k)$
  $(vk, sk) \xleftarrow{\$}$ SIGSETUP(1^k)
  $M_0, M_1 \xleftarrow{\$} \mathscr{A}_1(1^k)$
  return M_0, M_1
end function
function $\mathscr{B}_2^1(c)$
  $H \leftarrow$ MAKEH$_{pk}(c)$
  $\pi \xleftarrow{\$} S_2(\omega_{sim}, \tau, (c, H))$
  $\sigma \xleftarrow{\$}$ SIGN(c, H, π)
  return $\mathscr{A}_2(c, H, \pi, \sigma)$
end function
function MAKEH$_{pk=(X_1,X_2)}(c)$
  $r_1, r_2, r_3 \xleftarrow{\$} \mathbb{Z}_p$
  $H_1 \leftarrow X_1^{r_1} \cdot \prod_{i=1}^{|c|} c_{i,1}$
  $H_2 \leftarrow X_2^{r_2} \cdot \prod_{i=1}^{|c|} c_{i,2}$
  $H_3 \leftarrow g^{r_1+r_2} \cdot \prod_{i=1}^{|c|} c_{i,3}$
  return $H = (H_1, H_2, H_3)$
end function

Algoritmo 16 Algoritmo $\mathscr{B}^2$

function $\mathscr{B}^2(1^k)$
$(\omega_{sim}, \tau) \xleftarrow{\$} S_1(1^k)$
$(vk, sk) \xleftarrow{\$} \text{SIGSETUP}(1^k)$
$(pk, dk) \xleftarrow{\$} \text{ENCSETUP}(1^k)$
$b \xleftarrow{\$} \{0,1\}$
$M_0, M_1 \xleftarrow{\$} \mathscr{A}_1(1^k)$
$c \xleftarrow{\$} \text{SIMENCRYPT}_{pk}(M_b)$
$H \leftarrow \text{MAKEH}_{pk}(c)$
$\pi \xleftarrow{\$} S_2(\omega_{sim}, \tau, (c, H))$
$\sigma \xleftarrow{\$} \text{SIGN}(c, H, \pi)$
return $b = \mathscr{A}_2(c, H, \pi, \sigma)$
end function
function $\text{SIMENCRYPT}_{pk=(X_1,X_2)}(M)$
$r_{1,1}, \ldots, r_{1,k=|M|} \xleftarrow{\$} \mathbb{Z}_p$
$r_{2,1}, \ldots, r_{2,k=|M|} \xleftarrow{\$} \mathbb{Z}_p$
$c \leftarrow \{(X_1^{r_{1,i}}, X_2^{r_{2,i}}, g^{r_{1,i}+r_{2,i}} \cdot u_i^{m_i})\}_{i=1}^{k=|M|}$
return c
end function
function $\text{MAKEH}_{pk=(X_1,X_2)}(c)$
$r_1, r_2, r_3 \xleftarrow{\$} \mathbb{Z}_p$
$H_1 \leftarrow X_1^{r_1} \cdot \prod_{i=1}^{|c|} c_{i,1}$
$H_2 \leftarrow X_2^{r_2} \cdot \prod_{i=1}^{|c|} c_{i,2}$
$H_3 \leftarrow g^{r_1+r_2} \cdot \prod_{i=1}^{|c|} c_{i,3}$
return $H = (H_1, H_2, H_3)$
end function

Es admisible, sin pérdida de generalidad, suponer que la probabilidad dentro de este valor absoluto es mayor a $1/2$, pues para todo adversario tal que esta probabilidad sea menor a $1/2$, un adversario que conteste exactamente lo contrario poseerá la misma ventaja y una probabilidad mayor a $1/2$. Así,

$$\begin{aligned} \mathrm{Adv}^{\text{IND-CPA}}(\mathscr{B}^1) &= 2 \cdot \mathbb{P}[\mathscr{B}^2(1^k) = 1] - 1 \\ \implies \mathbb{P}[\mathscr{B}^2(1^k) = 1] &= \frac{1}{2}\mathrm{Adv}^{\text{IND-CPA}}(\mathscr{B}^1) + \frac{1}{2} \end{aligned}$$

De la misma forma, $\mathbb{P}[\mathscr{B}^2(1^k) = 1]$ equivale a $\mathbb{P}[\mathscr{B}^3(1^k) = 1]$, como se define en el algoritmo 17.

Algoritmo 17 Algoritmo $\mathscr{B}^3$

function $\mathscr{B}^3(1^k)$
  $(\omega_{sim}, \tau) \xleftarrow{\$} S_1(1^k)$
  $(vk, sk) \xleftarrow{\$} \textsc{SigSetup}(1^k)$
  $(pk, dk) \xleftarrow{\$} \textsc{EncSetup}(1^k)$
  $b \xleftarrow{\$} \{0,1\}$
  $M_0, M_1 \xleftarrow{\$} \mathscr{A}_1(1^k)$
  $(c, H, \pi) \xleftarrow{\$} \textsc{Encrypt}_{pk,vk}(M_b)$
  $\sigma \xleftarrow{\$} \textsc{Sign}(c, H, \pi)$
  return $b = \mathscr{A}_2(c, H, \pi, \sigma)$
end function

En $\mathscr{D}$, ENCRYPT utiliza σ_{sim} al momento de construir las demostraciones. La diferencia entre $\mathscr{B}^2$ y $\mathscr{B}^3$ es que este último utiliza PROVE para generar la demostración π en vez de S_2. Sin embargo, la distribución que sigue π es idéntica en ambos casos debido a que el sistema de demostración de Groth-Sahai posee la propiedad Zero Knowledge componible. De esta forma, la equivalencia entre probabilidades se cumple. Por otro lado, considérese el algoritmo $\mathscr{B}^4$, definido en el algoritmo 18.

Este algoritmo reproduce perfectamente el experimento IND-CPA para el esquema propuesto. De esta forma (y, de nuevo considerando que sin pérdida de generalidad puede removerse el valor absoluto),

$$\begin{aligned} \mathrm{Adv}^{\text{IND-CPA}}(\mathscr{A}) &= |2 \cdot \mathbb{P}[\mathscr{B}^4(1^k) = 1] - 1| \\ \implies \mathbb{P}[\mathscr{E}(1^k) = 1] &= \frac{1}{2}\mathrm{Adv}^{\text{IND-CPA}}(\mathscr{A}) + \frac{1}{2} \end{aligned}$$

Algoritmo 18 Algoritmo $\mathscr{B}^4$

function $\mathscr{B}^4(1^k)$
$\omega_{crs} \xleftarrow{\$} \text{PROOFSETUP}(1^k)$
$(vk, sk) \xleftarrow{\$} \text{SIGSETUP}(1^k)$
$(pk, dk) \xleftarrow{\$} \text{ENCSETUP}(1^k)$
$b \xleftarrow{\$} \{0,1\}$
$M_0, M_1 \xleftarrow{\$} \mathscr{A}_1(1^k)$
$(c, H, \pi) \xleftarrow{\$} \text{ENCRYPT}_{pk,vk}(M_b)$
$\sigma \xleftarrow{\$} \text{SIGN}(c, H, \pi)$
return $b = \mathscr{A}_2(c, H, \pi, \sigma)$
end function

De esta forma, es útil considerar un último adversario, $\mathscr{B}^5$, esta vez con respecto al experimento de indistinguibilidad entre las diferentes configuraciones del esquema de Groth-Sahai, como se define en el algoritmo 19.

Algoritmo 19 Adversario $\mathscr{B}^5$

function $\mathscr{B}^5(\omega)$
$(vk, sk) \xleftarrow{\$} \text{SIGSETUP}(1^k)$
$(pk, dk) \xleftarrow{\$} \text{ENCSETUP}(1^k)$
$b \xleftarrow{\$} \{0,1\}$
$M_0, M_1 \xleftarrow{\$} \mathscr{A}_1(1^k)$
$(c, H, \pi) \xleftarrow{\$} \text{ENCRYPT}_{pk,vk}(M_b)$
$\sigma \xleftarrow{\$} \text{SIGN}(c, H, \pi)$
$b' \xleftarrow{\$} \mathscr{A}_2(c, H, \pi, \sigma)$
return $b = b'$
end function

La ventaja de este adversario se expresa de la siguiente manera:

$$\text{Adv}^{\omega}(\mathscr{B}^5) = |\mathbb{P}[\mathscr{B}^5(\omega) = 1 \mid \omega = \omega_{crs}] - \mathbb{P}[\mathscr{B}^5(\omega) = 1 \mid \omega = \omega_{sim}]|$$

Por la forma de los algoritmos $\mathscr{B}^3$, $\mathscr{B}^4$ y $\mathscr{B}^5$, se cumple:

$$|\mathbb{P}[\mathscr{B}^5(\omega) = 1 \mid \omega = \omega_{crs}] - \mathbb{P}[\mathscr{B}^5(\omega) = 1 \mid \omega = \omega_{sim}]| = |\mathbb{P}[\mathscr{B}^4(1^k) = 1] - \mathbb{P}[\mathscr{B}^3(1^k) = 1]|$$

Utilizando las equivalencias antes mostradas,

$$\begin{aligned} |\mathbb{P}[\mathscr{B}^4(1^k) = 1] - \mathbb{P}[\mathscr{B}^3(1^k) = 1]| &= |\frac{1}{2}\text{Adv}^{\text{IND-CPA}}(\mathscr{A}) + \frac{1}{2} - \frac{1}{2}\text{Adv}^{\text{IND-CPA}}(\mathscr{B}^1) - \frac{1}{2}| \\ &= |\frac{1}{2}\text{Adv}^{\text{IND-CPA}}(\mathscr{A}) - \frac{1}{2}\text{Adv}^{\text{IND-CPA}}(\mathscr{B}^1)| \end{aligned}$$

Despejando, se llega a una expresión que relaciona las ventajas de $\mathscr{A}$, $\mathscr{B}$ y $\mathscr{F}$:

$$\begin{aligned} \mathrm{Adv}^{\omega}(\mathscr{B}^5) &= |\frac{1}{2}\mathrm{Adv}^{\text{IND-CPA}}(\mathscr{A}) - \frac{1}{2}\mathrm{Adv}^{\text{IND-CPA}}(\mathscr{B}^1)| \\ \implies \mathrm{Adv}^{\omega}(\mathscr{B}^5) &\geq \frac{1}{2}\mathrm{Adv}^{\text{IND-CPA}}(\mathscr{A}) - \frac{1}{2}\mathrm{Adv}^{\text{IND-CPA}}(\mathscr{B}^1) \\ \implies \mathrm{Adv}^{\text{IND-CPA}}(\mathscr{A}) &\leq \mathrm{Adv}^{\text{IND-CPA}}(\mathscr{B}^1) + 2 \cdot \mathrm{Adv}^{\omega}(\mathscr{B}^5) \end{aligned}$$

El supuesto *DLIN* implica que $\mathrm{Adv}^{\omega}(\mathscr{B}^5)$ es un valor pequeño, según [38]. Finalmente, es necesario acotar el valor de $\mathrm{Adv}^{\text{IND-CPA}}(\mathscr{B}^1)$. Para ello, es útil considerar el siguiente esquema de encriptación $\mathscr{SBE}$ (*single-bit encryption*, encriptación de un bit) como se define en el algoritmo 20.

Algoritmo 20 Esquema $\mathscr{SBE}$

function KEYGEN(1^k)
  $x_1, x_2 \xleftarrow{\$} \mathbb{Z}_p$
  $dk \leftarrow (x_1, x_2)$
  $pk \leftarrow (X_1, X_2) = (g^{x_1}, g^{x_2})$
  return (pk, dk)
end function

function ENCRYPT$_{pk=(X_1,X_2)}(m)$
  $r_1 \xleftarrow{\$} \mathbb{Z}_p$
  $r_2 \xleftarrow{\$} \mathbb{Z}_p$
  $c \leftarrow (X_1^{r_1}, X_2^{r_2}, g^{r_1+r_2} \cdot m)$
  return c
end function

function DECRYPT$_{dk=(x_1,x_2)}(c = (c_1, c_2, c_3))$
  return $(c_3 \cdot (c_1^{1/x_1} c_2^{1/x_2})^{-1})$
end function

Un adversario $\mathscr{B}^1$ para el esquema $\mathscr{BLE}$ puede verse como el adversario $\mathscr{B}^6$, definido en el algoritmo 21, para $\mathscr{SBE}$ (donde se permite que el adversario realice múltiples preguntas al oráculo LOR).

Es claro que $\mathscr{B}^6$ gana el juego IND-CPA (con múltiples consultas) para $\mathscr{SBE}$ si y sólo si $\mathscr{B}$ gana el juego IND-CPA para $\mathscr{BLE}$. Además, puede demostrarse [40, §10.2.2] que, si la ventaja de un adversario $\mathscr{B}^6$ que realiza q consultas al oráculo LOR es $f(k)$, entonces existe un adversario $\mathscr{B}^7$ en el sentido IND-CPA estándar, de forma que:

$$Adv^{\text{IND-CPA}}(\mathscr{B}^7) \leq f(k) \cdot q$$

Finalmente, se puede utilizar esto para acotar de forma definitiva la ventaja IND-CPA de cualquier

Algoritmo 21 Adversario $\mathscr{B}^6$ para $\mathscr{SBE}$

function $\mathscr{B}^6(1^k)$
  $M^0, M^1 \xleftarrow{\$} \mathscr{B}_1^1(1^k)$
  $L \leftarrow |M^0|$ ▷ $|M^0| = |M^1|$
  $u_0, ..., u_L \xleftarrow{\$} \mathbb{G}$
  for $i = 1, ..., L$ **do**
    $c_i \xleftarrow{\$} \text{LoR}(u_i^{M_i^0}, u_i^{M_i^1})$
  end for
  return $\mathscr{B}_2^1(c_1||...||c_L)$
end function

adversario para el esquema propuesto:

$$\begin{aligned} & \text{Adv}^{\text{IND-CPA}}(\mathscr{A}) \leq \text{Adv}^{\text{IND-CPA}}(\mathscr{B}^1) + 2 \cdot \text{Adv}^{\omega}(\mathscr{B}^5) \\ \implies & \text{Adv}^{\text{IND-CPA}}(\mathscr{A}) \leq \text{Adv}^{\text{IND-CPA}}(\mathscr{B}^7) \cdot |M| + 2 \cdot \text{Adv}^{\omega}(\mathscr{B}^5) \end{aligned}$$

□

B .2 Infalsificabilidad del esquema de firma subyacente

La infalsificabilidad del esquema de firmas utilizado en la solución propuesta queda demostrada por el siguiente lema:

Lema 4. *El esquema propuesto es un esquema de firmas sobre textos cifrados infalsificable.*

Demostración. Sea $\mathscr{SC}$ el esquema de firmas sobre textos cifrados propuesto, siendo $\mathscr{S}$ el esquema de firmas de Waters modificado. Se tiene que este último cumple la propiedad de no falsificabilidad ante ataques de texto extendido escogido, bajo el supuesto CDH. Es necesario acotar la cantidad

$$\text{Adv}^{\text{c-uf}}(\mathscr{A})$$

para cualquier adversario de tiempo polinomial $\mathscr{A}$. Asumiendo que un adversario $\mathscr{A}$ es capaz de romper la no falsificabilidad de $\mathscr{SC}$, se construirá un adversario $\mathscr{B}$ capaz de atacar exitosamente el esquema de firmas de Waters modificado.

Antes de comenzar, es necesario notar que, dado que $\mathscr{B}$ genera los parámetros del esquema de commitments utilizado para las demostraciones de conocimiento de M, g^{r_1}, g^{r_2}, puede extraer estos valores.

$\mathscr{B}$ realiza la simulación de $\mathscr{SC}$ para $\mathscr{A}$ como sigue:

SETUP(1^k): se ejecuta SETUP$^{\mathscr{S}}(1^k)$, del cual se obtienen los parámetros $(p,\mathbb{G},\mathbb{G}_T,e,g,h,u_0,...,u_l)$. $\mathscr{B}$ genera los parámetros del sistema de demostración de Groth-Sahai de modo de poder extraer los valores contenidos en los commitments.

EKEYGEN(1^k): se escogen $x_1,x_2 \xleftarrow{\$} \mathbb{Z}_p$, que definen $dk=(x_1,x_2)$ y $pk=(X_1,X_2)=(g^{x_1},g^{x_2})$.

SKEYGEN(1^k): se obtiene $vk_{\mathscr{S}}$ de la función SKEYGEN$^{\mathscr{S}}$. Se fija $vk=Y=vk_{\mathscr{S}}$.

ESIGN$_{vk,pk}(C=(c,\Pi_r,\Pi_M)$:

- Si c no es consistente con las demostraciones Π_r,Π_M, se retorna $\perp$
- De otro modo, se puede extraer M de Π_M, ya que contiene commitments sobre cada bit de M. También es posible extraer $Y_1=Y^{r_1}$ e $Y_2=Y_2^r$ de los commitments de Π_r. Además, puede calcular, usando dk, $R_1=c_1^{1/x_1}, R_2=c_2^{1/x_2}$. Así, $\mathscr{B}$ puede realizar la consulta SIGN$^{\mathscr{S}}_{sk_{\mathscr{S}}}(M,R_1,R_2,Y_1,Y_2)$ a su oráculo de firmas, obteniendo $\sigma'=(\sigma'_1,\sigma'_2,\sigma'_3,\sigma'_4)=(sk_{\mathscr{S}}\cdot(\mathrm{F}(M)R_1R_2)^s, g^{-s}, R_1^{-s}, R_2^{-s})$. Se agrega (vk,M) al conjunto SM, y se retorna:

$$\sigma=\begin{pmatrix} \sigma_1=\sigma_3'^{-x_1}=g^{sr_1x_1}=Y_1^{sr_1}, & \sigma_2=\sigma_4'^{-x_2}=g^{sr_2x_2}=Y_2^{sr_2}, \\ \sigma_3=\sigma_1'=sk_{\mathscr{S}}\cdot\mathrm{F}(M)^s\cdot g^{s(r_1+r_2)}, & \sigma_4=\sigma_2'^{-x_1}=g^{sx_1}=X_1^s, \\ \sigma_5=\sigma_2'^{-x_2}=g^{sx_2}=X_2^s, & \sigma_6=\sigma_2'^{-1}=g^s \end{pmatrix}$$

Luego de una cantidad de consultas polinomial, $\mathscr{A}$ emite, con probabilidad no despreciable, una firma σ^* sobre un texto cifrado realeatorizable $C^*=(c^*,H^*,\Pi_r,\Pi_M)$ válido. Del mismo modo que en ESIGN, $\mathscr{B}$ puede extraer M de la demostración Π_M contenida en C. Para que la falsificación sea válida, debe cumplirse que $M\neq\perp$ y $(M,vk)\notin SM$. Además, y análogamente a lo realizado en ESIGN, pueden extraerse (R_1,R_2,Y_1,Y_2). De esta forma, $\mathscr{B}$ genera la siguiente falsificación de una firma de Waters modificada:

$$\Sigma_{\mathscr{B}}=(\Sigma_1,\Sigma_2,\Sigma_3,\Sigma_4)=(\sigma_3^*,\sigma_6^{*-1},\sigma_1^{*-1/x_1},\sigma_2^{*-1/x_2})$$

Para ver que esta es una firma de Waters modificada válida, es necesario notar que σ^* cumple las siguientes propiedades, ya que es una firma sobre textos cifrados reencriptables válida:

$$\begin{aligned} e(\sigma_3^*,g)&=e(h,Y)\cdot e(c_3^*,\sigma_6^*) \\ e(\sigma_1^*,X_1)&=e(c_1^*,\sigma_4^*) \\ e(\sigma_2^*,X_2)&=e(c_2^*,\sigma_5^*) \\ e(\sigma_1^*,g)&=e(c_1^*,\sigma_6^*) \\ e(\sigma_2^*,g)&=e(c_2^*,\sigma_6^*) \end{aligned}$$

Para que $\Sigma_{\mathscr{B}}$ sea una firma válida, es necesario que se cumplan las siguientes ecuaciones:

$$\begin{aligned} e(Y,h) &= e(g,\Sigma_1)\cdot e(\mathrm{F}(M)R_1R_2,\Sigma_2) \\ e(g,\Sigma_3) &= e(\Sigma_2,R_1) & e(g,\Sigma_4) &= e(\Sigma_2,R_2) \\ e(R_1,Y) &= e(g,Y_1) & e(R_2,Y) &= e(g,Y_2) \end{aligned}$$

Se mostrará cómo se cumple cada una de estas ecuaciones. Para la primera de estas ecuaciones, es conveniente demostrar una equivalencia que servirá para rematar la demostración. La presencia de la demostración Π_M en C^* implica, por definición:

$$\begin{aligned} & e(c_3^*,Y) = e(u_0\prod_{i=1}^{|M|}u_i^{m_i},Y)\cdot e(Y_1Y_2,g) \\ \iff\ & e(c_3^*,Y) = e(\mathrm{F}(M),Y)\cdot e(Y_1Y_2,g) \end{aligned}$$

Pero $e(Y_i,X_i) = e(c_i^*,Y)$. Luego, como por definición $R_i^{x_i} = c_i^*$ y $X_i = g^{x_i}$, se tiene que:

$$\begin{aligned} & e(Y_i,X_i) = e(c_i^*,Y) \\ \implies\ & e(Y_i,g^{x_i}) = e(R_i^{x_i},Y) \\ \implies\ & e(Y_i,g) = e(R_i,Y) \\ \implies\ & e(Y_1Y_2,g) = e(R_1R_2,Y) \end{aligned}$$

Con esto, se puede reescribir la ecuación anterior:

$$\begin{aligned} & e(c_3^*,Y) = e(\mathrm{F}(M),Y)\cdot e(Y_1Y_2,g) \\ \iff\ & e(c_3^*,Y) = e(\mathrm{F}(M),Y)\cdot e(R_1R_2,Y) \\ \iff\ & e(c_3^*,Y) = e(\mathrm{F}(M)R_1R_2,Y) \\ \implies\ & e(c_3^*,x) = e(\mathrm{F}(M)R_1R_2,x)\forall x\in\mathbb{G} \end{aligned} \tag{6.1}$$

La última implicancia se tiene del hecho de que el grupo $\mathbb{G}$ al que pertenecen c_3^* e Y tiene orden primo, por lo que todo elemento es generador: en particular, todo elemento de $\mathbb{G}$ puede ser expresado como Y^r, para un r particular.

Esta equivalencia permite demostrar que la primera de las ecuaciones de comprobación de la firma falsificada se cumple. Debido a que σ^* es una firma válida, se cumple:

$$\begin{aligned} & e(h,Y)\cdot e(c_3^*,\sigma_6^*) = e(\sigma_3^*,g) \\ \iff & e(Y,h) = e(g,\sigma_3^*)\cdot e(c_3^*,\sigma_6^{*-1}) \end{aligned}$$

De esta forma:

$$\begin{aligned} & e(g,\Sigma_1)\cdot e(\mathrm{F}(M)R_1R_2,\Sigma_2) = e(Y,h) \\ \iff\ & e(g,\sigma_3^*)\cdot e(\mathrm{F}(M)R_1R_2,\sigma_6^{*-1}) = e(Y,h) \\ \iff\ & e(g,\sigma_3^*)\cdot e(\mathrm{F}(M)R_1R_2,\sigma_6^{*-1}) = e(g,\sigma_3^*)\cdot e(c_3^*,\sigma_6^{*-1}) \\ \iff\ & e(\mathrm{F}(M)R_1R_2,\sigma_6^{*-1}) = e(c_3^*,\sigma_6^{*-1}) \end{aligned}$$

La identidad (6.1) concluye la demostración, al fijarse $x = \sigma_6^{*-1}$.

La demostración de las demás ecuaciones resulta ser más directa. En particular, pueden hacerse de a pares. Para la segunda y tercera ecuación, se tiene, para $i = \{1,2\}$, $j = \{3,4\}$, respectivamente:

$$\begin{aligned} & e(g, \Sigma_j) = e(\Sigma_2, R_i) \\ \iff & e(g, \sigma_i^{*-1/x_i}) = e(\sigma_6^{*-1}, R_i) \\ \iff & e(\sigma_i^*, g) = e(\sigma_6^*, R_i^{x_i}) \\ \iff & e(c_i^*, \sigma_6^*) = e(\sigma_6^*, R_i^{x_i}) \end{aligned}$$

Este último paso es posible dado que $e(\sigma_i^*, g) = e(c_i^*, \sigma_k^*)$, pues σ^* es una firma válida. Se concluye, entonces, a partir de la conmutatividad del mapa e y de la definición de R_i.

Las últimas dos ecuaciones (con $i = \{1,2\}$ respectivamente) quedan demostradas por lo siguiente:

$$\begin{aligned} & e(R_i, Y) = e(g, Y_i) \\ \iff & e(c_i^*, Y) = e(g, Y_i) \\ \iff & e(c_i^{*1/x_i}, Y) = e(g, Y_i) \\ \iff & e(c_i^*, Y) = e(g^{x_i}, Y_i) \\ \iff & e(c_i^*, Y) = e(X_i, Y_i) \end{aligned}$$

La última ecuación se cumple debido a la presencia de la demostración Π_r, que asegura que $e(Y_i, X_i) = e(c_i^*, Y)$ (debido a la propiedad de consistencia perfecta).

De esta forma, $\sigma_{\mathscr{B}}$ constituye una firma válida para (M, R_1, R_2, Y_1, Y_2), con lo que $\mathscr{B}$ produce una falsificación válida para $\mathscr{S}$.

Con esto, se tiene que $\text{Adv}^{\text{uf-cma}}(\mathscr{B}) = \text{Adv}^{\text{c-uf}}(\mathscr{A})$. Como el esquema de firmas de Waters posee infalsificabilidad, se concluye que $\text{Adv}^{\text{uf-cma}}(\mathscr{B})$ es pequeña; así, $\text{Adv}^{\text{c-uf}}(\mathscr{A})$ también lo será, con lo que se concluye la demostración. □