Centro de Investigación y de Estudios Avanzados
del Instituto Politécnico Nacional

Unidad Zacatenco

Departamento de Computación

Diseño de un protocolo para votaciones electrónicas basado en firmas a ciegas definidas sobre emparejamientos bilineales

Tesis que presenta

M. en C. María de Lourdes López García

para obtener el Grado de

Doctora en Ciencias en Computación

Director de la Tesis

Dr. Francisco José Rambó Rodríguez Henríquez

México, D.F. **Junio 2011**

Proverbios 19:21
El corazón humano genera muchos proyectos
pero al final prevalecen los designios del Señor.

A mis amados Padres Juan y Rufina
por quienes siento profundo amor, respeto y admiración.

María de Lourdes López García

Agradecimientos

Proverbios 19:23
Se alista el caballo para el día de la batalla
pero la victoria depende del Señor.

Gracias a Dios por brindarme la oportunidad de cerrar este ciclo. Culmino mis estudios de doctorado llevándome en el corazón las múltiples experiencias con las que aprendí a ser una mejor persona a través de la gente con la que tuve el privilegio de interactuar durante este periodo.

Gracias a mi familia, icono fundamental para alcanzar mis metas, en especial a mi tía Carmen López quien ha sido una persona maravillosa e incondicional.

Agradezco a mi director de tesis el Dr. Francisco Rodríguez Henríquez por las enseñanzas obtenidas a lo largo de mis estudios, por su carácter estricto en lo académico y por lo comprensivo en situaciones que se me presentaron difíciles de sobrellevar.

Agradezco a los Doctores Debrup Chakraborty, Luis Gerardo de la Fraga, Miguel Ángel León Chávez y Moisés Salinas Rosales por sus valiosos comentarios en la revisión de mi tesis.

Agradezco especialmente al Dr. Carlos Coello Coello y al Dr. José Guadalupe Rodríguez por el apoyo brindado.

Al Dr. Santiago Domínguez, al Ing. Jose Luis Flores, al Ing. Arcadio Morales y a las señoras Felipa Rosas, Erica Ríos y Flor Córdova de quienes aprecio mucho su eficiencia, amabilidad y disponibilidad para apoyarme en todo lo necesario.

Agradezco la colaboración del Dr. Luis Julian Domínguez y de mis compañeros Luis Martínez, Jorge González, Jonathan Taverne y Armando Faz por su colaboración al facilitarme las diferentes bibliotecas criptográficas utilizadas en mi tesis.

Agradezco a la Sra. Sofia Reza y a William de la Cruz quienes son mi luz y mi alegría. A Sofi por que es una amiga incondicional, mi consejera y a William por enseñarme a ver y a vivir la vida de diferente manera. Con todo mi corazón agradezco a Dios haberlos conocido.

Agradezco a todas las personas que cruzaron mi camino en este periodo, en especial a mi amigo y compañero de generación Antonio López a quien considero una gran persona y deseo para él todo el éxito. Así como a Magaly, Ismael, Nidia, Daniel, Arturo, Cuauhtemoc, Gregorio, Alfredo, Paquito, Gabriel, Eduardo, Julio, Saul y Amilcar y las muy platicadoras Lil y Laurita mil gracias por hacer tan amena mi estancia en el Cinvestav.

Agradezco al Consejo Nacional de Ciencia y Tecnología (CONACyT) por el apoyo económico que me fue otorgado a partir de su programa de becas y de los proyectos SEP-CONACyT 90543, SEP-CONACyT 60240 y UC MEXUS para llevar a buen término mis estudios de doctorado.

Al Centro de Investigación y de Estudios Avanzados del IPN (CINVESTAV-IPN) por permitirme ser parte de esta institución y por el apoyo económico otorgado en la finalización de mis estudios.

Resumen

En la actualidad, el avance en la tecnología ha permitido pasar de sistemas de elección manual a esquemas automatizados de votación electrónica, por medio de dispositivos electrónicos o de Internet. De tal forma que tanto las entidades electorales como los votantes pueden desarrollar una elección de manera eficiente, con la facilidad en la emisión del voto o la contabilidad de los mismos en tiempos muy cortos. Sin embargo, los esquemas de votación electrónica deben cumplir requerimientos que garanticen la seguridad del voto, el anonimato del votante y la confiabilidad de los resultados. En este documento, se presenta un esquema de votación electrónica que realiza un protocolo entre el votante y dos entidades electorales de manera eficiente y con un número mínimo de operaciones criptográficas por fase.

El esquema propuesto en esta tesis utiliza como herramientas criptográficas principales, la firma a ciegas propuesta por Boldyreva en 2003 y la firma corta propuesta por Boneh en 2001. Ambas firmas usan la criptografía basada en emparejamientos y una función especial denominada *map-to-point*. La boleta generada por el esquema consiste en únicamente una firma a ciegas y una firma corta con sus respectivos mensajes, lo que ocasiona un paso de mensajes de tamaño mínimo entre las entidades, un mejor rendimiento y mayor seguridad que los esquemas de votación electrónica basados en firmas a ciegas propuestos en la literatura.

Abstract

Nowadays, technological achievements have made possible to go from manual elections to electronic voting systems which rely on electronic devices on the Internet. In this way, both electoral entities and voters can hold an efficient election, where a vote can be easy to cast and the result of the election can be available in a short period of time. However, electronic voting schemes must fulfill several requirements, which guarantee vote's integrity, voter's anonymity, and reliable results. In this thesis, we present an electronic voting scheme which performs the communication among voters and electoral entities with the minimum number of cryptographic operations per phase.

The proposed scheme in this thesis uses as main cryptographic tools the blind signature scheme proposed by Boldyreva in 2003 and the short signature scheme proposed by Boneh in 2001. Both signatures based their security on pairing-based cryptography and a special function called *map-to-point*. Our scheme generates ballots which consist of just one blind signature, one short signature and two messages, hence, our scheme gets a better performance and more security than other electronic voting schemes based on blind signatures reported in the literature.

Índice general

Índice de figuras

Índice de tablas

Capítulo 1

Introducción

1.1. Planteamiento del problema

Los avances en la tecnología y las técnicas criptográficas han hecho posible considerar a las votaciones electrónicas como una alternativa factible para las elecciones tradicionales. Los medios de comunicación como el Internet y los dispositivos electrónicos como las computadoras personales, los teléfonos celulares, tarjetas inteligentes, etc., facilitan la captura, transmisión y recepción del voto, permitiendo así un sistema de elecciones más cómodo para los votantes y más eficiente para las autoridades electorales.

Sin embargo, la eficiencia y la comodidad que ofrecen los esquemas de votación electrónica abren la puerta a distintos problemas de los ya existentes en las votaciones tradicionales. Por ejemplo, la identificación del votante no es tan simple en las votaciones electrónicas como lo es en las convencionales, si se considera que el votante debe ser identificado antes de votar y a su vez su identidad debe ser anónima al momento de emitir el voto.

Otro problema es la auditoria de la elección. En las votaciones tradicionales, el resguardo de las boletas electorales impresas ofrece una garantía de escrutinio. Si existieran dudas o desconfianza en los resultados finales, bastaría con realizar un recuento de votos para verificar la validez del resultado final. De manera electrónica, el voto es capturado y transmitido hasta la urna electoral electrónica para ser contabilizado al final del periodo de votación, sin alguna comprobación contundente que asegure al votante que su voto fue correctamente recibido y contabilizado.

Lo que indican los problemas mencionados es que, considerando que Internet es un canal inseguro de comunicación, son necesarios los esquemas de seguridad para votaciones electrónicas que permitan ofrecer la seguridad y la privacidad requeridas en todo proceso electoral. Sin embargo, no siempre es obvia la forma de alcanzar tales características a un precio razonable, debido al hecho de que cuando un proceso electoral se lleva a cabo, los mecanismos que aseguran tanto la seguridad como la privacidad pueden ser demasiado costosos para los administradores (entidades electorales) por un lado, e

inconvenientes para los usuarios (votantes) por el otro. En este sentido, un sistema de votación electrónica no puede ser completamente automatizado. No obstante, mediante el uso de herramientas criptográficas, es posible satisfacer los requisitos de seguridad más importantes asociados a los esquemas de votación electrónica.

En esta tesis estamos interesados en los sistemas de votación electrónica los cuales utilizan Internet para transmitir los votos. Informalmente, este tipo de sistemas se puede definir como sigue.

Definición 1.1.1. *Un sistema electrónico de votación por Internet es un sistema de elección que genera boletas electorales electrónicas, que permite a los votantes emitir su voto desde un dispositivo electrónico y transmitirlo por Internet hacia la urna electoral también electrónica donde será depositado y contabilizado al término de la jornada electoral.*

De manera general, un sistema de votación electrónica por Internet debe cubrir todos los requisitos funcionales del proceso electoral, así como los servicios de seguridad necesarios para protegerse de potenciales ataques provenientes de la red. Algunos de los requisitos esenciales son los siguientes.

- ▷ *Autenticación*: sólo los votantes incluidos en el padrón electoral serán autorizados para emitir su voto.
- ▷ *Anonimato y no coerción*: nadie debe ser capaz de determinar el valor del voto ni de vincularlo con el votante.
- ▷ *Unicidad*: ningún votante debe votar más de una sola vez.
- ▷ *Integridad*: los votos no pueden ser modificados, olvidados o borrados sin ser detectado.
- ▷ *Verificación y auditoría*: debe ser posible verificar que al final del proceso electoral, todos los votos fueron contados correctamente, demostrando así la honorabilidad del sistema.

1.2. Motivación

El uso de Internet como medio de comunicación para llevar a cabo un sistema de votación se ha traducido en la posibilidad de fraude electoral debido a los posibles ataques de seguridad inherentes a dicha red. Ello implica que en los sistemas de votación por Internet se requieran protocolos de seguridad que permitan garantizar dos objetivos en primera instancia contradictorios entre sí, los cuales consisten en la autenticación del votante por un lado, pero sin por esto vulnerar su derecho al voto secreto, por el otro.

En la literatura se han propuesto una variedad de protocolos criptográficos con la finalidad de evitar el fraude electoral y satisfacer el anonimato del votante. Grosso modo, estos protocolos pueden ser divididos en dos clases principales: basados en firmas a ciegas y basados en funciones homomórficas [89]. Sin embargo, podemos considerar también protocolos que utilizan redes mezcladas (mixnets) [51, 46] o pruebas de conocimiento nulo [25].

En los esquemas de votación, las funciones homomórficas proveen una herramienta para obtener el conteo de los votos sin descifrarlos [76]. Algunos de los esquemas propuestos basados en funciones homomórficas pueden ser consultados en [69, 71, 72].

Las firmas a ciegas en los esquemas de votación electrónica permiten la obtención del voto, sin identificar al votante que lo ha emitido. Entre los esquemas publicados recientemente basados en firmas a ciegas, el lector interesado puede consultar los siguientes [48, 53, 23].

A pesar de la variedad de los protocolos propuestos de votaciones electrónicas usando, tanto funciones homomórficas como firmas a ciegas, ninguno de ellos ha logrado ofrecer una solución completa a los múltiples problemas o conflictos que surgen, como son la relación entre el votante y su voto, votar en más de una ocasión y la auditoría del sistema, entre otros.

Además de considerar la seguridad en los sistemas de elección electrónica también es importante tomar en cuenta su eficiencia. Actualmente, los esquemas propuestos basados en firmas a ciegas están definidos en grupos tanto multiplicativos como aditivos, es decir, lo que usan números enteros en un campo finito primo o usando puntos que satisfagan una curva elíptica. En el contexto de la criptografía basada en emparejamientos, los cuales usan tanto grupos aditivos como multiplicativos, los protocolos de seguridad para votaciones electrónicas basados en firmas a ciegas, antes de este trabajo de tesis, no habían sido explorados todavía.

1.3. Objetivos

Esta tesis tiene como objetivo general, *desarrollar un esquema seguro de votación electrónica el cual satisface los principales requerimientos de seguridad, utilizando como principal herramienta criptográfica una firma a ciegas basada en emparejamientos bilineales.*
Los objetivos particulares son los siguientes:

- Analizar los esquemas de votación electrónica existentes en la literatura.
- Analizar e implementar esquemas de firma digital.
- Analizar e implementar esquemas de firma a ciegas.

- Desarrollar un esquema de votación electrónica utilizando como bloque principal un esquema de firma a ciegas.
- Implementar el esquema de votación electrónica propuesto.
- Analizar eficiencia y seguridad del esquema de votación propuesto.

1.4. Contribuciones

1. La principal contribución en esta tesis es un esquema de seguridad para votaciones electrónicas que utiliza como bloques principales la firma a ciegas propuesta por Boldyreva [11] y la firma corta propuesta por Boneh et al. [12]. El esquema cumple con los requisitos de seguridad de las votaciones electrónicas y además es más eficiente que los esquemas propuestos recientemente publicados en la literatura.

2. Un esquema de firma a ciegas definido sobre curvas elípticas basado en la versión con números enteros de la firma a ciegas de Camenisch et al. [15], con su correspondiente análisis de seguridad y eficiencia.

3. La implementación y el análisis de seguridad y eficiencia de siete esquemas de firma a ciegas, de acuerdo al grupo donde se encuentran definidos, es decir, grupos multiplicativos (criptografía sobre enteros), grupos aditivos (criptografía sobre curvas elípticas) o la combinación de ambos (criptografía basada en emparejamientos). Cada esquema fue considerado con un nivel de seguridad de 128 bits.

4. Análogo al análisis e implementación de varios esquemas de firmas a ciegas, en esta tesis se presenta la implementación, análisis de seguridad y eficiencia para cinco esquemas de firma digital.

5. A través de una búsqueda del estado del arte en las votaciones electrónicas, se eligieron e implementaron tres esquemas de este tipo publicados recientemente en la literatura. Tales esquemas fueron comparados con nuestra propuesta, resultando esta última la más segura y eficiente.

1.5. Metodología

Para alcanzar el objetivo de esta tesis, se analiza el modelo de capas de los esquemas de votación electrónica basados en firmas a ciegas y para nuestro caso en particular utilizando emparejamientos bilineales.

Como se muestra en la Figura 1.1, de abajo hacia arriba, en la primera capa se tiene la aritmética de campos finitos, que incluye las operaciones básicas como la suma, multiplicación, inversión y la exponenciación modular.

La capa siguiente corresponde a la curvas elípticas, donde las operaciones de suma, doblado y bisección de puntos son requeridas para el cáculo de la multiplicación escalar, que para este trabajo es la operación principal en esta capa.

La capa tres corresponde a los emparejamientos bilineales, donde se encuentran las funciones de emparejamiento de *Weil* y *Tate*, y versiones posteriores de éste último tales como *ate*, *R-ate* y *ate óptimo* entre otros.

La capa de firmas contiene tanto las firmas digitales como las firmas a ciegas, ambas utilizan la función de emparejamiento principalmente en la fase de verificación y la multiplicación escalar en las distintas fases para la producción de la firma.

Por último se tiene la capa de los esquemas de votación electrónica, los cuales apoyan su seguridad en la buena elección de los esquemas de firma y consiguen una implementación eficiente de acuerdo a los campos definidos para establecer la aritmética en las capas anteriores. Cabe mencionar, que antes de este trabajo, esquemas de votación electrónica basados en emparejamientos bilineales no habían sido propuestos todavía.

Figura 1.1: Modelo de capas de un esquema de votación electrónica

1.6. Organización del documento

El resto del documento está organizado como sigue. En el Capítulo 2 se presenta el fundamento matemático y los algoritmos utilizados para la implementación de las operaciones requeridas en las dos primeras capas del modelo presentado en la Figura 1.1.

En el Capítulo 3 son analizados e implementados cinco esquemas de firma digital, los cuales están divididos de acuerdo a la operación del grupo donde realizan su aritmética. Además, se presenta una comparación general, de seguridad y de eficiencia entre ellos. Por último, es seleccionado uno de los esquemas de firma para ser implementado en el esquema de votación electrónica propuesto.

En el Capítulo 4, se enlistan la definición, propiedades y requerimientos de las firmas a ciegas. También son analizados e implementados nueve esquemas divididos de la misma forma que las firmas digitales y se comparan la eficiencia y la seguridad de todos ellos. Además, es presentado un esquema novedoso de firma a ciegas definido sobre curvas elípticas, el cual cumple con todos los requerimientos de un esquema de firma a ciegas. Por último, es seleccionado el mejor esquema para su implementación en la propuesta de votación electrónica.

En el Capítulo 5, son presentados la definición, tipos y requerimientos de los esquemas de votación electrónica por Internet. Además, se analizan e implementan tres esquemas de votación basados en firmas a ciegas propuestos recientemente en la literatura. Finalmente, se presenta un esquema de votación electrónica que combina la criptografía sobre curvas elípticas y la criptografía basada en emparejamientos.

En el Capítulo 6 es presentada la contribución final de esta tesis, la cual comprende un protocolo de seguridad para votaciones electrónicas basado en firmas a ciegas utilizando emparejamientos bilineales y se realiza el análisis de seguridad y eficiencia del mismo. Por último, se realiza una comparación general, de eficiencia y de seguridad entre el esquema propuesto y los tres esquemas mostrados en el capítulo anterior.

Por último, en el Capítulo 7 son enunciadas las conclusiones y el trabajo futuro que podría surgir de esta tesis.

Capítulo 2

Primitivas criptográficas básicas

En este capítulo se presenta el fundamento matemático y los algoritmos de las operaciones criptográficas básicas que utilizan los esquemas criptográficos presentados a lo largo de este documento.

2.1. Preliminares

A lo largo de la historia, se ha buscado mantener la privacidad en la comunicación entre dos o más personas. Proteger la información ante un tercero es el objetivo de la criptografía, donde se han propuesto muchos esquemas de seguridad para distintas aplicaciones.

El uso de técnicas criptográficas tiene como propósito prevenir las fallas de seguridad en un sistema electrónico. La seguridad en general, se refiere al cumplimiento de los servicios de seguridad que se listan a continuación [79]:

- *Confidencialidad* garantiza que la información privada pueda ser accedida únicamente por las entidades autorizadas.
- *Integridad de los datos* se refiere a la protección de los datos de tal manera que no puedan ser modificados, destruidos o extraviados de una manera maliciosa o accidental.
- *Autenticación* es un servicio relacionado con la identificación de entidades o de datos. La autenticación de una entidad es la confirmación de su identidad, es decir, una comprobación de que es quien dice ser. La autenticación de datos se refiere a la validez de los datos, lo que implica la integridad de los mismos.
- *No rechazo* asegura que el remitente de cierta información no pueda rechazar/negar su transmisión o contenido y que el receptor no pueda negar su recepción o contenido.

Con la finalidad de cumplir los cuatro puntos anteriores tanto en la teoría como en la práctica, las técnicas o primitivas criptográficas deben ser evaluadas respecto a varios criterios, tales como:

- El *nivel de seguridad* que es una cota inferior en términos de las operaciones necesarias para alcanzar un objetivo, usando los mejores métodos conocidos. En otras palabras, es el número mínimo de operaciones requeridas para romper la seguridad de una primitiva dada.

- La *funcionalidad* indica la necesidad de combinar varias técnicas criptográficas para cumplir todos o la mayoría de los servicios de seguridad en un cripto-sistema.

- Los *métodos de operación* son las diferentes formas en las que las primitivas pueden ser implementadas, dependiendo de su funcionalidad.

- El *rendimiento* se refiere a la eficiencia de la primitiva, en términos de tiempo de ejecución, tamaño del código en el caso de bibliotecas de software y área del circuito en caso de implementaciones en hardware, para un modo de operación en particular.

- La *facilidad en la implementación* requiere el mínimo de dificultad para poner en práctica una primitiva, ya sea en un ambiente de software o de hardware.

De manera general la criptografía se divide en dos categorías: de llave secreta y de llave pública. El objetivo fundamental en ambas clases es mantener la comunicación segura entre dos entidades comúnmente llamadas *Alicia* y *Beto* a través de un canal inseguro el cual puede ser interceptado por algún oponente denominado *Eva* para obtener información valiosa.

Los cripto-esquemas de llave secreta se caracterizan por el hecho de usar una llave secreta para ocultar y descubrir la información que se desea proteger. Los cripto-esquemas de llave pública utilizan un par de llaves, una pública y otra privada, ambas asignadas a cada usuario del sistema y normalmente la llave pública se deriva de la llave privada.

En la criptografía moderna, los cripto-esquemas de llave pública son ampliamente usados para generar firmas digitales. El concepto de firma digital es análogo al de una firma autógrafa, pero tiene el servicio adicional de proteger la información de alteraciones intencionales de alguna entidad maliciosa.

En esta tesis, los esquemas de firma digital son analizados en su seguridad y eficiencia. Por tal motivo, este capítulo está dedicado a la implementación de las operaciones criptográficas básicas, considerando la explicación detallada de las propiedades, funcionalidad y seguridad de los esquemas de firma digital en los capítulos posteriores.

Debido a que los esquemas de llave pública pueden ser definidos en grupos tanto multiplicativos como aditivos o ambos, es decir, usando números enteros en un campo

finito primo o usando puntos que satisfagan una curva elíptica. Las operaciones criptográficas básicas presentadas son las operaciones predominantes en las que descansa la complejidad de los esquemas de firma. Así, la operación principal sobre los grupos multiplicativos es la exponenciación modular, para los grupos aditivos es la multiplicación escalar y en el contexto de la criptografía basada en emparejamientos, es decir, grupos aditivos transformados a grupos multiplicativos, es la función de emparejamiento bilineal.

Particularmente, presentaremos el fundamento matemático y los algoritmos que fueron utilizados para obtener los costos de implementación de la multiplicación escalar y la función de emparejamiento y con esto reportar la eficiencia en número de ciclos de cada cripto-esquema de llave pública mencionado en esta tesis.

2.2. Exponenciación modular

La exponenciación modular es una operación fundamental en la teoría de números computacional y es implementada en la mayoría de los sistemas criptográficos. Es de la forma $m^e \bmod n$ y significa multiplicar m por sí misma tanta veces según el valor de e y después obtener el resto respecto al módulo n.

El cálculo de la exponenciación modular en los cripto-esquemas de llave pública como los algoritmos de firma digital RSA [77] y DSA [67] significa multiplicar la base, la cual es un número grande, tantas veces lo indique el exponente. Por lo que es necesario implementar algoritmos de exponenciación eficientes.

Una manera inocente de desarrollar la exponenciación $m^e \bmod n$ es primero calculando $c = m \bmod n$ seguido de $c = c \times m \bmod n$, $e - 1$ veces. Por lo que se necesitan $e - 1$ multiplicaciones modulares, lo cual es altamente ineficiente.

Si se considera una secuencia de enteros $a_0, a_1, ...a_r$, con $a_0 = 1$ y $a_r = e$. La secuencia puede ser construida de tal manera que para todo k exista un par de índices $i, j < k$ tal que,

$$a_k = a_i + a_j$$

entonces la secuencia es una *cadena de adición* para e de longitud r.

Una cadena de adición para el exponente e es el algoritmo que calcula $m^e \bmod n$, comenzando con m^1 seguido de $m^{a_k} = m^{a_j} \times m^{a_i}$ y el número de multiplicaciones requeridas es r. Entonces, como una opción mejorada la exponenciación modular puede ser implementada calculando r multiplicaciones en lugar de $e - 1$.

Si la exponenciación modular es con un exponente fijo significa que diferentes números pueden ser elevados a una misma potencia. Por tanto, es conveniente obtener una cadena de adición para lograr mayor eficiencia. Por otro lado, si la exponenciación modular es con base fija significa que un número fijo puede ser elevado a diferentes potencias, por tanto, pueden realizarse cálculos previos (precómputo) para algunas potencias y éstas pueden ser almacenadas y re-utilizadas cuantas veces sea necesario por los algoritmos.

En general, los costos de implementación para la exponenciación modular han sido reportados en muchas ocasiones usando diferentes algoritmos y han sido implementados en distintas infraestructuras. En nuestro caso, para implementar la exponenciación modular se utilizó la biblioteca MIRACL (Multiprecision Integer and Rational Arithmetic C/C++ Library) [81], la cual contiene las primitivas necesarias para diseñar esquemas criptográficos.

2.3. Multiplicación escalar sobre curvas elípticas

2.3.1. Conceptos básicos

Una *curva elíptica* E definida sobre un campo finito $\mathbb{F}_{p^m}$, donde p es un número primo y m un entero positivo, está definida por la ecuación general de Weierstrass y es de la forma:

$$E : y^2 + a_1xy + a_3y = x^3 + a_2x^2 + a_4x + a_6 \tag{2.1}$$

con $a_1, a_2, a_3, a_4, a_6 \in \mathbb{F}_{p^m}$ y $\Delta \neq 0$, donde Δ es el discriminante de E y está definido como sigue:

$$\left.\begin{array}{lcl} \Delta & = & -d_2^2d_8 - 8d_4^3 - 27d_6^2 + 9d_2d_4d_6 \\ d_2 & = & a_1^2 + 4a_2 \\ d_4 & = & 2a_4 + a_1a_3 \\ d_6 & = & a_3^2 + 4a_6 \\ d_8 & = & a_1^2a_6 + 4a_2a_6 - a_1a_3a_4 + a_2a_3^2 - a_4^2 \end{array}\right\}$$

Sea E una curva elíptica definida sobre un campo finito $K = \mathbb{F}_{p^m}$, donde p es la *característica* de E. Podemos considerar a la ecuación 2.1 de forma simplificada de acuerdo a la característica del campo.

1. Si la característica del campo es distinta de 2 o 3 entonces la ecuación de la curva es de la forma:

   $$y^2 = x^3 + ax + b$$

   donde $a, b = \pm 0 \in \mathbb{F}_q$.

2. Si la característica es 2, tenemos dos casos a considerar. Si la ecuación de la curva es de la forma:

   $$y^2 + xy = x^3 + ax^2 + b$$

   donde $a, b \in \mathbb{F}_q$. La curva es llamada *ordinaria*. Si la ecuación de la curva es la siguiente:

   $$y^2 + cy = x^3 + ax + b$$

   donde $a, b, c \in \mathbb{F}_q$ y $a \neq 0$, entonces la curva es conocida como *supersingular*.

3. Si la característica es 3, igual que en el anterior, tenemos dos casos. La curva es *ordinaria* si la ecuación es de la forma:

$$y^2 = x^3 + ax^2 + b$$

donde $a, b \in \mathbb{F}_q$. de lo contrario la curva es *supersingular* si la ecuación es la siguiente:

$$y^2 = x^3 + ax + b$$

donde $a, b \in \mathbb{F}_q$.

Sea E una curva elíptica definida en $\mathbb{F}_q$, con $q = p^m$. Denotamos con $E(\mathbb{F}_q)$ al grupo de todos los puntos $(x, y) \in \mathbb{F}_q \times \mathbb{F}_q$ que satisfacen la Ecuación 2.1 denominados *puntos racionales*, con $x, y \in \mathbb{F}_q$. Sabemos que $E(\mathbb{F}_q)$ junto con el punto al infinito denotado como $\mathcal{O}$ forman un grupo aditivo abeliano, con $\mathcal{O}$ como elemento identidad.

La ley de grupo para la *suma* de puntos y el *doblado* de un punto está representada geométricamente en las Figuras 2.1 y 2.2. Sean $P = (x_1, y_1)$ y $Q = (x_2, y_2)$ dos puntos diferentes sobre E, la *suma* de P con Q es el punto R el cual es el reflejo respecto al eje x del punto que interseca a la curva con la línea que pasa a través de P y Q. El *doblado* de un punto P es el punto R el cual es el reflejo respecto al eje x del punto que interseca a la curva con la línea tangente que pasa por P.

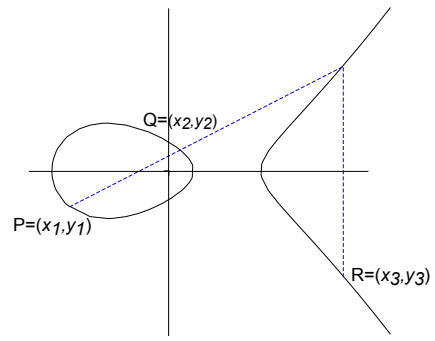

Figura 2.1: Suma: P+Q = R.

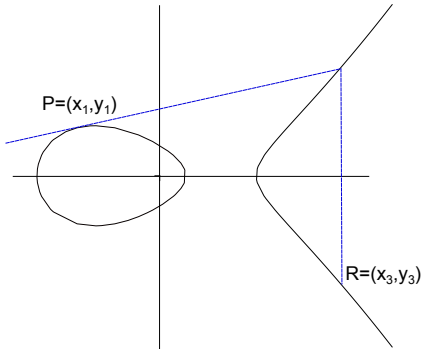

Figura 2.2: Doblado: P+P = R

Las fórmulas algebraicas para la ley de grupo son definidas de acuerdo a la curva elíptica E simplificada de Weierstrass. La Tabla 2.1 muestra la ley de grupo para la curvas $E(\mathbb{F}_p)$ y $E(\mathbb{F}_{2^m})$ definidas sobre un campo primo y un campo binario respectivamente:

Orden de la curva

Sea E una curva elíptica definida sobre $\mathbb{F}_q$. El número de puntos en $E(\mathbb{F}_q)$ denotado como $\#E(\mathbb{F}_q)$, se denomina el *orden de la curva* E sobre $\mathbb{F}_q$. El teorema Hasse & Weil provee una aproximación para $\#E(\mathbb{F}_q)$.

Curva elíptica ordinaria definida en un campo primo $K = \mathbb{F}_p$		
Negativo P	$-P = (x, -y)$	
Suma $R = P+Q$	$x_3 = \left(\frac{y_2-y_1}{x_2-x_1}\right)^2 - x_1 - x_2$	$y_3 = \left(\frac{y_2-y_1}{x_2-x_1}\right)(x_1 - x_3) - y_1$
Doblado $R = P+P$	$x_3 = \left(\frac{3x_1^2+a}{2y_1}\right)^2 - 2x_1$	$y_3 = \left(\frac{3x_1^2+a}{2y_1}\right)(x_1 - x_3) - y_1$
Curva elíptica ordinaria definida en un campo binario $K = \mathbb{F}_{2^m}$		
Negativo P	$-P = (x, x + y)$	
Suma $R = P+Q$	$x_3 = \left(\frac{y_1+y_2}{x_1+x_2}\right)^2 + \frac{y_1+y_2}{x_1+x_2} + x_1 + x_2 + a$	$y_3 = \left(\frac{y_1+y_2}{x_1+x_2}\right)(x_1+x_3)+x_3+y_1$
Doblado $R = P+P$	$x_3 = x_1^2 + \frac{b}{x_1^2}$	$y_3 = x_1^2 + \frac{x_1+y_1}{x_1}x_3 + x_3$

Tabla 2.1: Ley de Grupo para curvas definidas en los campos $K = \{\mathbb{F}_p$ y $\mathbb{F}_{2^m}\}$, con $P = (x_1, y_1), Q = (x_2, y_2)$ y $R = (x_3, y_3) \in E(K)$.

Teorema 2.3.1. *(Hasse & Weil)*
Sea E una curva elíptica definida sobre $\mathbb{F}_q$ entonces,

$$\#E(\mathbb{F}_q) = q + 1 - t,\ |t| \leq 2\sqrt{q}$$

donde t se denomina *traza del endomorfismo de Frobenius.* Dado que $2\sqrt{q}$ es relativamente pequeño a q, tenemos que $\#E(\mathbb{F}_q) \approx q$.

Orden del grupo

Sea E una curva elíptica en $\mathbb{F}_q$, $P \in E(\mathbb{F}_q)$ y r un entero positivo primo relativo de q. El *orden del grupo* $\mathbb{G}$ es el número de puntos generados por P y se denota como $\mathbb{G} = \langle P \rangle \in E(\mathbb{F}_q)$, donde $|\mathbb{G}_1| = r$ y $r|\#E(\mathbb{F}_q)$. P se denomina *punto base* de $\mathbb{G}$.

Definición 2.3.1. Sea E una curva elíptica sobre $\mathbb{F}_q$, d un número entero positivo y $P, Q \in E(\mathbb{F}_q)$. La *multiplicación escalar* es la operación que calcula el múltiplo $Q = dP$, definido como el punto resultante de sumar $P + P + \cdots + P$, d veces.

Existen diferentes técnicas para calcular la multiplicación escalar dP. La forma tradicional es expresando el escalar d en su representación binaria $(d_{n-1}, ..., d_0)$, $\sum_{i=0}^{n-1} d_i 2^i$ con $d_i \in \{0, 1\}$ para todo $i = 0, ..., n-1$, donde n es la longitud en bits de d, y utilizando el método de suma y doblado de puntos [37], con lo cual se consigue un costo

computacional de

$$(n-1)D + (\frac{n-1}{2})A$$

donde D y A representan las operaciones de doblado y suma de puntos, respectivamente [37].

Los esquemas criptográficos presentados en este documento utilizan distintos parámetros de dominio, por lo cual, es necesario mencionar por separado las técnicas utilizadas para el cálculo de la multiplicación escalar según los campos en donde se encuentra definida la curva.

2.3.2. Multiplicación escalar en $E(\mathbb{F}_p)$

Sea r un número primo, $E : y^2 = x^3 + b$ una curva elíptica ordinaria definida sobre un campo finito primo $\mathbb{F}_p$, donde el primo p cumple que $p \equiv 1 \bmod 4$ y $P \in E(\mathbb{F}_p)$ es un punto base de orden primo r. En esta tesis, La multiplicación escalar es desarrollada eficientemente utilizando el método GLV propuesto por Gallant et al. [34], el cual usa un endomorfismo definido como sigue:

Un endomorfismo sobre la curva E es una transformación racional $\phi : E \to E$ que satisface $\phi(\mathcal{O}) = \mathcal{O}$. Dado que E está definida en el campo primo $\mathbb{F}_p$ entonces ϕ es un homomorfismo de $E(\mathbb{F}_p)$ así como de $E(\mathbb{F}_{p^m})$ para alguna $m > 1$.

Si $\alpha \in \mathbb{F}_p$ es un elemento de orden 4 entonces la transformación $\phi : E \to E$ definida por $(x, y) \mapsto (-x, \alpha y)$ y $\mathcal{O} \mapsto \mathcal{O}$ es un endomorfismo en $\mathbb{F}_p$. Como P es un punto de orden r entonces ϕ actúa sobre $\mathbb{G}_1 = \langle P \rangle$ como una transformación multiplicativa λ, es decir, para todo $Q \in \mathbb{G}_1$ se cumple que $\phi(Q) = \lambda Q$, donde λ es un entero que satisface $\lambda^2 + 1 \equiv 0 \bmod r$.

La estrategia para calcular dP es descomponer a $d \in \mathbb{Z}_r$ en su *representación balanceada de dos longitudes* [37], es decir,

$$d = d_1 + d_2\lambda \bmod r$$

con $d_1, d_2 \in \mathbb{Z}$. El problema de encontrar d_1 y d_2 se resuelve usando el algoritmo extendido de Euclides para encontrar el máximo común divisor de r y λ. El algoritmo produce la secuencia $s_i r + t_i \lambda = w_i$, con $s_0 = 1, t_0 = 0, w_0 = r, s_1 = 0, t_1 = 1, w_1 = \lambda$, $w_i \geq 0$ y además l es el índice más grande para el cual $w_l \geq \sqrt{r}$. Una vez obtenida la secuencia $s_i r + t_i \lambda = w_i$, el Algoritmo 2.1 es utilizado para obtener los valores d_1 y d_2 tales que $|d_1| = |d_2| \approx \sqrt{r}$, es decir, la representación balanceada de dos longitudes para el escalar d.

Algoritmo 2.1: Representación balanceada de dos longitudes [37]

Entrada: r, λ, $d \in [0, r-1]$.
Salida: d_1, d_2 tales que $|d_1|, |d_2| \approx \sqrt{r}$.

1 Ejecutar el algoritmo extendido de Euclides para obtener la secuencia $s_i r + t_i \lambda = w_i$;
2 $(a_1, b_1) \leftarrow (w_{l+1}, -t_{l+1})$;
3 **if** $(w_l^2 + t_l^2) \leq (w_{l+2}^2 + t_{l+2}^2)$ **then**
4 $\quad (a_2, b_2) \leftarrow (w_l, t_l)$
5 **else**
6 $\quad (a_2, b_2) \leftarrow (w_{l+2}, t_{l+2})$
7 **end**
8 $c_1 = \lfloor b_2 d/r \rceil$; $c_2 = \lfloor -b_1 d/r \rceil$;
9 $d_1 = d - c_1 a_1 - c_2 a_2$;
10 $d_2 = -c_1 b_1 - c_2 b_2$;
11 **return** d_1, d_2

Teniendo la representación $d = d_1 + d_2 \lambda \bmod r$ es posible realizar una multiplicación simultánea usando el método de entrelazado (*interleaving*) [37].

El entrelazado es un método de multiplicación escalar simultánea que calcula $uP+vQ$ usando $[u_i P + v_i P]$ para $0 \leq i \leq n'-1$, los cuales son los puntos previamente calculados de acuerdo a la representación de los escalares u y v de n' bits de longitud. En el Algoritmo 2.2, el entrelazado se realiza con la expresión $d_1 P + d_2 \phi(P)$.

Para obtener mayor eficiencia, los escalares d_1 y d_2 son representados de la forma JSF (*Joint Sparse Form*) [37], con lo cual se espera obtener un número considerable de columnas en cero del arreglo de exponentes de (d_1, d_2),

$$(d_1, d_2) = \begin{pmatrix} d_{1,n'-1} & \dots & d_{1,1} & d_{1,0} \\ d_{2,n'-1} & \dots & d_{2,1} & d_{2,0} \end{pmatrix}$$

En el Algoritmo 2.2 se muestra la ventaja de utilizar la representación balanceada y la JSF del escalar d. Para lograr una mejor eficiencia se realiza el cálculo previo de los puntos $\pm P \pm \phi(P)$, con los cuales en la línea 6 se ahorran sumas cuando $d_{1,i} = 0$ o $d_{2,i} = 0$ según lo indicado en el arreglo de exponentes. El costo computacional del Algoritmo 2.2 para calcular $dP = d_1 P + d_2 \phi(P)$, con cuatro puntos calculados previamente, es aproximadamente de,

$$(2 + 0.5n')A + n'D$$

donde $|d| = n$, $|n'| \approx \frac{|n|}{2}$.

Algoritmo 2.2: Método de entrelazado con JSF [37]

Entrada: $d, P \in E(\mathbb{F}_p)$, r.
Salida: $Q = dP$.

```
1 Descomponer d en d_1 + d_2 λ mod r;
2 Obtener la representación JSF para d_1 y d_2 con |d_1| = |d_2| ≈ n';
3 Q ← O;
4 for i = n' − 1 downto 0 do
5     Q ← 2Q;
6     Q ← Q + [d_{1,i}P + d_{2,i}φ(P)];
7 end
8 return Q;
```

2.3.3. Multiplicación escalar en $E(\mathbb{F}_{p^2})$

Sea r un número primo, $E : y^2 = x^3 + b$ una curva elíptica ordinaria definida sobre un campo finito primo $\mathbb{F}_p$, donde el primo p cumple que $p \equiv 1 \bmod 4$ y $P \in E(\mathbb{F}_p)$ es un punto base de orden primo r. La curva E admite una torsión séxtica definida como [7],

$$\tilde{E}/\mathbb{F}_{p^2} : y^2 = x^3 + b'$$

sobre $\mathbb{F}_{p^2}$ con $r|\#\tilde{E}(\mathbb{F}_{p^2})$, $r^2 \nmid \#\tilde{E}(\mathbb{F}_{p^2})$.

Sea $\tilde{\mathbb{G}}_2 = \langle\tilde{Q}\rangle$, donde $\tilde{Q} \in \mathbb{F}_{p^2}$, $\mathbb{G}_2 = \langle Q\rangle$, donde $Q = \phi\tilde{Q}$ y $\phi : \tilde{E}(\mathbb{F}_{p^2}) \to E(\mathbb{F}_{p^k})$ es un homomorfismo eficiente y k es el grado de encajamiento de la curva tal que $r|p^k - 1$.

Sea π_p el endomorfismo de Frobenius en $E(\mathbb{F}_{p^k})$ definido como $\pi_p(x, y) = (x^p, y^p)$. Entonces $\psi : \mathbb{G}_2 \to \mathbb{G}_2$ es un homomorfismo sobre $\tilde{E}(\mathbb{F}_{p^2})$ tal que

$$\psi := \phi^{-1}\pi_p\phi$$

y además,

i) ψ es un homomorfismo en $\mathbb{G}_2$
ii) para todo $R \in \mathbb{G}_2$ se cumple que $\psi^k(R) - R = \mathcal{O}$ y $\psi^2(R) - t\psi(R) + pR = \mathcal{O}$.
iii) existe un único $\lambda \in \mathbb{Z}_r$ tal que $\lambda^k - 1 \equiv 0 \bmod r$ y $\lambda^2 - t\lambda + p \equiv 0 \bmod r$ cumpliendo que $\psi(R) = \lambda R$ para todo $R \in \mathbb{G}_2$.

Si se considera que ψ satisface $\psi^4 - \psi^2 + 1 = 0$ es posible utilizar el método GLV de 4 dimensiones [32, 33]. De tal manera que la multiplicación escalar dQ con $d \in \mathbb{Z}_r$ pueda calcularse eficientemente utilizando el endomorfismo ψ para representar a d de la forma

$$n \equiv d_0 + d_1\lambda + d_2\lambda^2 + d_3\lambda^3 \bmod r$$

donde $|d_i| \approx \sqrt[4]{r}$.

El problema de descomponer d es encontrar un vector x más cercano a $w = (d, 0, 0, 0)$ dentro del conjunto parcialmente ordenado L tal que

$$L = \{x \in \mathbb{Z}^4 : \sum_{i=0}^{3} x_i \lambda^i \equiv 0 \bmod r\}$$

Aplicando una reducción a L utilizando el método de reducción *weakPopov* [6, 27] se obtiene una nueva base B con $\lambda = T = t - 1 = 6x^2$ de acuerdo a los parámetros de la curva Barreto-Naehrig la cual será definida en la sección 2.4.3.

$$B = \begin{pmatrix} x+1 & x & x & -2x \\ 2x+1 & -x & -(x+1) & -x \\ 2x & 2x+1 & 2x+1 & 2x+1 \\ x-1 & 4x+2 & -(2x-1) & x-1 \end{pmatrix}$$

con la cual se puede calcular un vector $v \approx wB^{-1}$ de la forma [32]:

$$wB^{-1} = \left(\tfrac{d(2x^2+3x+1)}{r}, \tfrac{d(12x^3+8x^2+x)}{r}, \tfrac{d(6x^3+4x^2+x)}{r}, \tfrac{d(-2x^2-x)}{r} \right)$$

Haciendo un redondeo a $wB^{-1} \approx v$ (Babai [2]) se puede calcular un vector $x = vB = (x_0, x_1, x_2, x_3)$ más cercano a w y con esto obtener $u = w - x$ cuyas entradas son los coeficientes d_i necesarios para la descomposición del escalar d.

Teniendo la descomposición de $d \equiv d_0 + d_1\lambda + d_2\lambda^2 + d_3\lambda^3 \bmod r$, la multiplicación escalar dQ es calculada utilizando la multiplicación simultánea con el método de entrelazado y tomando la expansión w-NAF de los d_i con $i \in [0, ..., 3]$.

La forma NAF de un entero positivo d es una expresión tal que $d = \sum_{i=0}^{l-1} d_i 2^i$, donde $d_i \in \{0, \pm 1\}$, $d_{l-1} \neq 0$ donde l es la longitud en bits de la expresión NAF, la cual es única, tiene pocos dígitos diferentes de cero y $l < n$.

Una representación w-NAF del entero d es una expresión $d = \sum_{i=0}^{l-1} d_i 2^i$, donde cada coeficiente diferente de cero d_i es impar, $|d_i| < 2^{w-1}$, $d_{l-1} \neq 0$, a lo más se tienen w dígitos diferentes de cero y l es la longitud de la representación resultante.

Como se muestra en el Algoritmo 2.3, el método de entrelazado calcula $\sum_{j=1}^{v} d_j Q_j$, la representación w-NAF es usada en d_j y los puntos iQ para $2^{w_j - 1}$ son calculados y almacenados con antelación.

El costo computacional del Algoritmo 2.3 para calcular $dP = d_0 Q + d_1 \psi Q + d_2 \psi^2 Q + d_3 \psi^3 Q$, con 12 puntos calculados previamente, es aproximadamente de,

$$(3 + .37n')A + (1 + n')D$$

donde $|d| = n$ y $|n'| \approx \frac{|n|}{4}$ [37].

Algoritmo 2.3: Método de entrelazado con representación w-NAF [37]

Entrada: $v = 4$, enteros d_j, ventana w_j y los puntos $Q_j, 1 \leq j \leq v$.
Salida: $S = dP$.

```
1  Calcular iQ_j para cada i ∈ {1, 3, ..., 2^{w_j - 1} − 1}, 1 ≤ j ≤ v;
2  Obtener la representación w-NAF de d^j;
3  l = max{l_j : 1 ≤ j ≤ v} ;
4  Definir d_{j,i} = 0 para cada l_j ≤ i ≤ l, 1 ≤ j ≤ v;
5  S ← O;
6  for i = l − 1 downto 0 do
7      S ← 2S;
8      for j = 1 to 2 do
9          if d_{j,i} ≠ 0 then
10             if d_{i,j} > 0 then
11                 S ← S + d_{i,j} Q_j
12             else
13                 S ← S − d_{ij} Q_j
14             end
15         end
16     end
17 end
18 return S;
```

2.3.4. Multiplicación escalar en $E(\mathbb{F}_{2^m})$

Dada una curva elíptica $E : y^2 + xy = x^3 + ax^2 + b$ definida en un campo binario $\mathbb{F}_{2^m}$ y un punto base P de orden primo r, la multiplicación escalar dP se calcula utilizando el método de bisección y suma de puntos [37], con la representación w-NAF del escalar d.

Dados $P, Q \in \mathbb{F}_{2^m}$, una *bisección* reemplaza el doblado de un punto por una operación más eficiente que produce Q de P tal que $P = 2Q = (u, v)$. Ya que la bisección es la operación inversa del doblado, la idea esencial para la bisección es resolver λ para la coordenada x y finalmente para la coordenada y en las siguientes ecuaciones,

$$\begin{aligned} \lambda &= x + y/x \\ u &= \lambda^2 + \lambda + a \\ v &= x^2 + u(\lambda + 1) \end{aligned}$$

la representación λ de un punto Q se denota como (u, λ_Q) donde $\lambda_Q = u + \frac{v}{u}$.

Si d es representado usando una d' tal que cumpla,

$$d = \frac{d'_{t-1}}{2^{t-1}} + \cdots + \frac{d'_2}{2^2} + \frac{d'_1}{2} + d'_0 \bmod r$$

entonces $(d'_{t-1}, ..., d'_0)$ puede ser utilizado para desarrollar la multiplicación escalar,

$$dP = \sum_{i=0}^{t-1} \frac{d'_i}{2^i} P$$

usando el método de bisección y puede ser generalizado a su expresión w-NAF.

Sea $\sum_{i=0}^{t} d'_i 2^i$ la representación w-NAF de $2^{t-1} d \bmod n$, entonces:

$$d \equiv \sum_{i=0}^{t-1} \frac{d'_{t-1-i}}{2^i} + 2d'_t (\bmod r)$$

El Algoritmo 2.4 calcula la multiplicación escalar utilizando sólo bisecciones y suma de puntos. El costo computacional del Algoritmo 2.4 para calcular dP usando la bisección con 2^{w-1} puntos calculados previamente, es aproximadamente de 2 doblados + 6 sumas + $\frac{t}{5} - 4$ sumas con representación $\lambda + t$ bisecciones.

Algoritmo 2.4: Método de Bisección y suma [83]

Entrada: w, $P \in E(\mathbb{F}_{2^m})$ de orden r, $NAF_w(2^{t-1} d \bmod r)$.
Salida: $Q = dP$.

```
Q_i ← O para cada i ∈ I = {1, 3, ..., 2^{w-1} − 1}.;
if d'_t = 1 then
    Q_1 = 2P;
end
for i = t − 1 downto 0 do
    if d'_i > 0 then
        Q_{d'_i} ← Q_{d'_i} + P;
    else
        if d'_i < 0 then
            Q_{−d'_i} ← Q_{−d'_i} − P;
        end
    end
    P ← P/2;
end
Q ← Σ_{i∈I} iQ_i;
return Q;
```

2.4. Función de emparejamiento bilineal

Sea r un número primo, $\mathbb{G}_1$ y $\mathbb{G}_2$ dos grupos aditivos con identidad $\mathcal{O}$, $\mathbb{G}_T$ un grupo multiplicativo con identidad 1 y $|\mathbb{G}_1| = |\mathbb{G}_2| = |\mathbb{G}_T| = r$.

Definición 2.4.1. *Un emparejamiento bilineal sobre los grupos* $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T)$ *es la transformación*

$$\hat{e} : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$$

que satisface las siguientes condiciones [10]:

- Bilinealidad. Para cada $P, P' \in \mathbb{G}_1, Q, Q' \in \mathbb{G}_2$ tenemos que $\hat{e}(P + P', Q) = \hat{e}(P, Q)\hat{e}(P', Q)$ y $\hat{e}(P, Q + Q') = \hat{e}(P, Q)\hat{e}(P, Q')$

- No degenerado.

  Para cada $P \in \mathbb{G}_1$, con $P \neq \mathcal{O}$, existe un $Q \in \mathbb{G}_2$ tal que $\hat{e}(P, Q) \neq 1$.

  Para cada $Q \in \mathbb{G}_2$, con $Q \neq \mathcal{O}$, existe un $P \in \mathbb{G}_1$ tal que $\hat{e}(P, Q) \neq 1$.

  $\hat{e}(P, Q) = 1$, si $P = \mathcal{O}$ o $Q = \mathcal{O}$.

- e puede ser eficientemente calculada.

Cuando $\mathbb{G}_1 = \mathbb{G}_2$, se dice que el emparejamiento es *simétrico*. En caso contrario, se denomina *asimétrico*. En la práctica, los emparejamientos simétricos son definidos sobre curvas elípticas supersingulares mientras que los asimétricos se definen sobre curvas ordinarias [17].

En el estado del arte de los emparejamientos bilineales se encuentra una diversidad de funciones, todas ellas basadas en dos propuestas iniciales, el *emparejamiento de Weil* [64] y el *emparejamiento de Tate* [29].

A continuación se describen los objetos y operadores básicos que son los bloques necesarios para construir una función de emparejamiento.

2.4.1. Divisores

Sea $K = \mathbb{F}_q$ un campo finito y $E(K)$ una curva elíptica definida sobre K. Para cada punto $P \in E(\overline{K})$ se define un símbolo formal (P).

Definición 2.4.2. Un *divisor* D es una combinación lineal de cada símbolo con coeficientes enteros,

$$D = \sum_j a_j(P),\ a_j \in \mathbb{Z}$$

Un divisor, por tanto, es un elemento del grupo libre abeliano generado por (P) y el grupo de divisores se denota como $Div(E)$.

El grado y la suma de un divisor se definen como:

$$\begin{aligned} deg(\sum_j a_j(P_j)) &= \sum_j a_j \in \mathbb{Z} \\ sum(\sum_j a_j(P_j)) &= \sum_j a_j P_j \in E(\overline{K}) \end{aligned}$$

la función *sum* usa la ley de grupo de E para sumar los puntos que están dentro de los símbolos.

Los divisores de grado cero forman un subgrupo de $Div(E)$ y se denota como $Div^0(E)$. Debido a que $sum([P] - [\mathcal{O}]) = P$, la función *sum* es un homomorfismo sobreyectivo definido como $sum : Div^0(E) \to E(\overline{K})$.

Supongamos que E está dada por la ecuación $y^2 = x^3 + ax + b$, una función sobre E es una *función racional*

$$f(x, y) \in \overline{K}[x, y]$$

que está definida por lo menos para un punto en $E(\overline{K})$.

Si f no puede ser evaluada el punto $(0, 0)$ puede ser siempre transformada tal que el resultado $0/0$ no exista y en $(0, 0)$ resulte un valor único determinado en $\overline{K} \cup \{\mathcal{O}\}$.

Si al evaluar f en el punto $P \in E(\mathbb{F}_q)$ toma el valor 0, entonces se dice que f tiene un *cero*. Por el contrario, si resulta el punto al infinito $\mathcal{O}$ entonces se dice que f tiene un *polo*. Una función racional esta determinada únicamente por sus ceros y polos.

La función conocida como *uniformador* y denotada como u_P es una función que permite la evaluación de un punto P sobre la función racional f tal que el resultado sea diferente de cero o de infinito. Así, la función $f(x, y)$ puede ser reescrita de la siguiente forma:

$$f = u_P^n g$$

donde $g(P)$ es diferente de cero y de infinito y $n \in \mathbb{Z}$ es el orden de la función f en P y se denota como $ord_P(f)$.

Definición 2.4.3. Si f es una función en E y no es cero, el *divisor de* f se define como:

$$div(f) = \sum_{P \in E(\overline{K})} ord_P(f)(P) \in Div(E)$$

y además,

1. f tiene un número finito de ceros y polos.

2. $deg(div(f)) = 0$.

3. Si f no tiene ni ceros ni polos entonces f es una función constante.

Sea E una curva elíptica y D un divisor en E de grado cero, entonces existe una función f en E con $div(f) = D$, si y solo si $sum(D) = \mathcal{O}$. D se denomina *divisor principal*.

Dos divisores D_1 y D_2 son equivalentes y se denota como $D_1 \sim D_2$ si se cumple que $D_1 - D_2 = div(f)$ es un divisor principal.

El grupo de divisores principales es un subgrupo de los divisores de grado cero, entonces la función sum es un isomorfismo definido como,

$$sum : Div^0(E)/\{\text{divisores principales}\} \rightarrow E(\overline{K})$$

En la siguiente sección mostraremos la importancia de los divisores en la construcción de la función de emparejamiento bilineal.

2.4.2. Función de Miller

Sean E una curva elíptica definida sobre un campo finito $\mathbb{F}_q$, $P \in E(\mathbb{F}_q)$, r un número primo, $E[r]$ el subgrupo de puntos generados por P de orden r y $f_{r,P}$ una función racional con divisor $r(P) - r(\mathcal{O})$. Para cada $i \geq 1$, tenemos que $f_{i,P}$ es una función cuyo divisor se obtiene como sigue,

$$div(f_{i,P}) = i(P) - (iP) - (i-1)(\mathcal{O})$$

Como se observa en la Figura 2.3, para cada par de enteros i, j, ℓ es la línea que une a iP con jP y v la línea vertical que pasa por $iP + jP$, entonces f_{i+j} se define como,

$$f_{i+j,P} = f_{i,P} \cdot f_{j,P} \cdot \frac{\ell}{v}$$

Sea $r = (r_t, ..., r_1, r_0)$ la representación binaria de r. La función $f_{i,P}$ puede ser eficientemente calculada utilizando el método de suma y doblado y puede ser utilizado para calcular el emparejamiento de Tate, como se muestra en el Algoritmo 2.5.

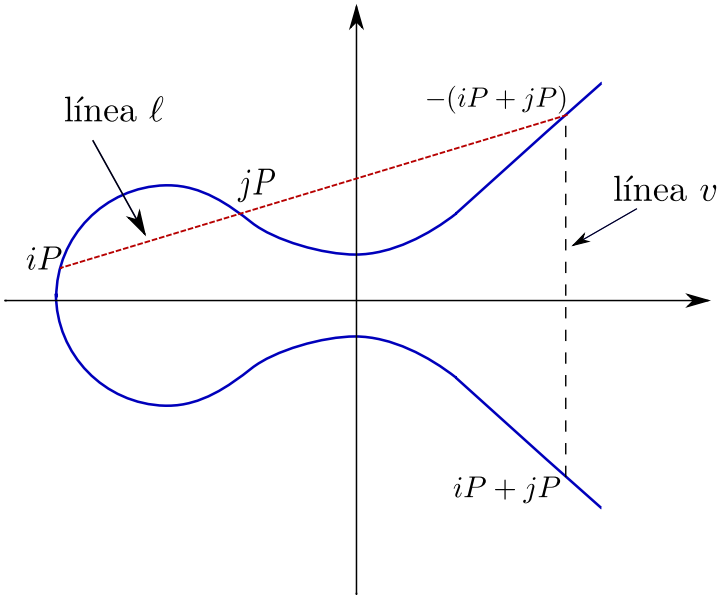

Figura 2.3: Suma de divisores en una curva elíptica

Algoritmo 2.5: Algoritmo de Miller [64]

Entrada: $P, R, r = (r_t ... r_1 r_0)$.
Salida: $f = \langle P, Q \rangle_r$ y la expansión binaria de r

```
1  T ← P; f ← 1;
2  for i = t − 1 downto 0 do
3      Calcular las ecuaciones de las líneas l1 y l2 resultantes del doblado
       de T;
4      T ← 2T;
5      f ← f^2 · l1(Q + R)l2(R)/(l2(Q + R)l1(R));
6      if r_i = 1 then
7          Calcular las ecuaciones de las líneas l1 y l2 resultantes de la
           suma de T con P;
8          T ← T + P;
9          f ← f · l1(Q + R)l2(R)/(l2(Q + R)l1(R));
10     end
11 end
12 return f
```

Barreto et al. en [5] realizaron una mejoras al algoritmo de Miller y obtuvieron mayor eficiencia del emparejamiento de Tate definido en curvas ordinarias definidas para campos primos, y curvas supersingulares definidas para campos binarios y ternarios.

Debido a que $\mathbb{F}_q^*$ es cíclico de orden $q-1$, las $(q-1)/r-$ésimas potencias dan el isomorfismo $\mathbb{F}_q^*/(\mathbb{F}_q^*)^r \to \mu_r$. Entonces se puede definir,

$$e(P,Q) : (P,Q)^{(q-1)/r}$$

Lo que garantiza que el emparejamiento de P y Q sea no degenerado, es decir, que el resultado del emparejamiento sea un valor único. A esta operación se le denomina *exponenciación final*.

Informalmente hablando, podemos decir que el cálculo de la función de emparejamiento se divide en el cómputo de la función de Miller y de la exponenciación final.

A partir de las mejoras reportadas por Barreto et al., diversos autores han propuesto modificaciones tanto al algoritmo de Miller como a la exponenciación final para ganar eficiencia en calculo de la función de emparejamiento. Los emparejamientos *ate* [40], *R-ate* [21] y *ate optimo* [88, 90, 9], son versiones modificadas del emparejamiento de Tate, con el propósito de reducir el número de operaciones que requiere la función de Miller.

Dado que existen en la literatura diversas propuestas para aumentar la eficiencia en el cálculo de los emparejamientos, en este documento se utiliza la función de emparejamiento *ate* óptimo [88] que como se indica en [9], es el más eficiente debido a que el número de iteraciones en la función de Miller es menor y consume menos tiempo de procesamiento.

2.4.3. Emparejamiento asimétrico *ate* óptimo sobre curvas BN

Sea E una curva elíptica ordinaria definida sobre un campo finito primo $\mathbb{F}_p$ denominada *curva Barreto-Naehrig* (BN) [7]. Las curvas BN están definidas por la ecuación:

$$E : y^2 = x^3 + b$$

con $b \neq 0$, grado de encajamiento $k = 12$ y el orden primo $\#E(\mathbb{F}_p) = r$. El primo p, el orden de la curva r y la traza de Frobenius t_r de la curva están parametrizadas como sigue [9],

$$\begin{aligned} p(z) &= 36z^4 + 36z^3 + 24z^2 + 6z + 1 \\ r(z) &= 36z^4 + 36z^3 + 18z^2 + 6z + 1 \\ t_r(z) &= 6z^2 + 1 \end{aligned}$$

donde $z \in \mathbb{Z}$ es un entero arbitrario tal que $p = p(z)$ y $r = r(z)$ son primos.

Sea $E[r]$ el subgrupo de puntos de torsión de E y $\pi_p : E \to E$ el endomorfismo de Frobenius dado por $\pi_p(x, y) = (x^p, y^p)$. Definimos,

$$\begin{aligned} \mathbb{G}_1 &= E[r] \cap Ker(\pi_p - [1]) = E(\mathbb{F}_p)[r] \\ \mathbb{G}_2 &= E[r] \cap Ker(\pi_p - [p]) \subseteq E(\mathbb{F}_{p^{12}})[r] \\ \mathbb{G}_T &= \langle \mu_r \rangle \subset \mathbb{F}^*_{p^{12}} \end{aligned}$$

donde $\mathbb{G}_T$ es el subgrupo de r-raíces de unidad de $\mathbb{F}_{p^{12}}$.

Las curvas BN admiten una torsión séxtica lo que significa que muchos cálculos pueden ser restringidos a la extensión cuadrática $\mathbb{F}_{p^2}$ en el grupo $\mathbb{G}_2$.

Sea E una curva BN con una torsión séxtica $\tilde{E}$ sobre $\mathbb{F}_{p^2}$. Sea $\tilde{E}/\mathbb{F}_{p^2} : y^2 = x^3 + b'$, con $r|\#\tilde{E}(\mathbb{F}_{p^2})$ y $r^2 \nmid \#\tilde{E}(\mathbb{F}_{p^2})$. Sea $\tilde{\mathbb{G}}_2 = \langle \tilde{Q} \rangle$, donde $\tilde{Q} \in \mathbb{F}_{p^2}$. Sea $\mathbb{G}_2 = \langle Q \rangle$, donde $Q = \phi\tilde{Q}$ y $\phi : \tilde{E}(\mathbb{F}_{p^2}) \to E(\mathbb{F}_{p^{12}})$ es un homomorfismo eficiente.
Entonces, se pueden redefinir los grupos de la siguiente manera:

$$\begin{aligned} \mathbb{G}_1 &= \langle P \rangle \in E(\mathbb{F}_p) \\ \mathbb{G}_2 &= \langle Q \rangle \in \tilde{E}(\mathbb{F}_{p^2}) \\ \mathbb{G}_T &= \langle \mu_r \rangle \subset \mathbb{F}^*_{p^{12}} \end{aligned}$$

Sean los grupos abelianos aditivos $\mathbb{G}_1 = \langle P \rangle \in E(\mathbb{F}_p)$ y $\mathbb{G}_2 = \langle Q \rangle \in \tilde{E}(\mathbb{F}_{p^2})$ y el grupo abeliano multiplicativo $G_T = \langle \mu \rangle \in \mathbb{F}^*_{p^{12}}$, todos ellos de orden r.

El emparejamiento *ate óptimo* a_{opt} es la transformación,

$$a_{opt} : \mathbb{G}_2 \times \mathbb{G}_1 \to \mathbb{G}_T$$

tal que

$$a_{opt}(Q, P) = (f_{6z+2,Q}(P) \cdot l_{[6z+2]Q+\pi_p(Q)}(P) \cdot l_{[6z+2]Q+\pi_p(Q),-\pi_p^2(Q)}(P))^{(p^{12}-1)/r}$$

donde l_{Q_i,Q_j} es la ecuación de la línea vertical que pasa por $Q_i, Q_j \in \mathbb{G}_2$ y π_p^2 denota la segunda aplicación del operador de Frobenius.

El Algoritmo 2.6 muestra el cálculo del emparejamiento *ate* óptimo. El objetivo principal del ciclo de Miller (líneas 3-13) es, primero construir la función racional $f_{6z+2,Q}$ asociada al punto Q y al entero $6z+2$ y posteriormente evaluarla en el punto P. Después, en las líneas 14-16, se evalúan las funciones $l_{[6z+2]Q,\pi_p(Q)}(P)$ y $l_{[6z+2]Q+\pi_p(Q),-\pi_p^2(Q)}(P)$, que se refieren a las líneas que cruzan por los puntos $[6z+2]Q$ y $\pi_p(Q)$ para la primera función y $[6z+2]Q + \pi_p(Q)$ y $-\pi_p^2(Q)(P)$ para la segunda. Finalmente, en la línea 17 se calcula la exponenciación final.

Algoritmo 2.6: Emparejamiento *ate* óptimo sobre curvas BN [9].

Entrada: $P \in \mathbb{G}_1$ and $Q \in \mathbb{G}_2$.
Salida: $a_{\text{opt}}(Q, P)$.

Escribir $s = 6z + 2$ as $s = \sum_{i=0}^{L-1} s_i 2^i$, con $s_i \in \{-1, 0, 1\}$;
$T \leftarrow Q$, $f \leftarrow 1$;
for $i = L - 2$ *to* 0 **do**
    $f \leftarrow f^2 \cdot l_{T,T}(P)$;
    $T \leftarrow 2T$;
    if $s_i = -1$ **then**
        $f \leftarrow f \cdot l_{T,-Q}(P)$; $T \leftarrow T - Q$
    else
        if $s_i = 1$ **then**
            $f \leftarrow f \cdot l_{T,Q}(P)$; $T \leftarrow T + Q$
        end
    end
end
$Q_1 \leftarrow \pi_p(Q)$; $Q_2 \leftarrow \pi_{p^2}(Q)$;
$f \leftarrow f \cdot l_{T,Q_1}(P)$; $T \leftarrow T + Q_1$;
$f \leftarrow f \cdot l_{T,-Q_2}(P)$; $T \leftarrow T - Q_2$;
$f \leftarrow f^{(p^{12}-1)/r}$;
return f;

2.5. Función especial *map-to-point*

La función H_1 es una función picadillo especial denominada *map-to-point*. Como su nombre lo indica, transforma una cadena de longitud arbitraria hacia un punto sobre una curva elíptica y se define como sigue. Sea $\mathbb{G}_1$ un grupo abeliano aditivo,

Definición 2.5.1. *La función picadillo denominada map-to-point es una transformación de la forma,*

$$H_1 : \{0, 1\}^* \rightarrow \mathbb{G}_1$$

Idealmente, H_1 debe cumplir con las propiedades de una función hash la cuales se abordarán en el capítulo 3.

2.5.1. $H_1 : \{0, 1\}^* \rightarrow \mathbb{G}_1 \in E(\mathbb{F}_p)$

Sea $E : y^2 = x^3 + b$ una curva elíptica ordinaria BN, definida sobre un campo primo $\mathbb{F}_p$, el punto base $P \in E(\mathbb{F}_p)$ y el grupo aditivo abeliano $\mathbb{G}_1 = \langle P \rangle$ de orden r.

H_1 es construida usando una función picadillo estándar, como por ejemplo SHA-2 [68], para la coordenada x tal que $x \in \mathbb{F}_p$, para después obtener la coordenada y, calculando $\sqrt{x^3 + b} = u$. Una vez obtenidas las coordenadas (x, y) debe verificarse que el punto recién generado pertenece al grupo $\mathbb{G}_1$. Si no es el caso, se buscaran nuevas coordenadas hasta encontrar un punto que satisfaga la pertenencia del grupo.

Como puede observarse en el Algoritmo 2.7, la parte fundamental para obtener un punto en $\mathbb{G}_1$ se encuentra en la línea 4, es decir, calcular una raíz cuadrada.

Müller en [66] propuso un algoritmo que combina el símbolo de Legendre [91] y la función de Lucas [45] para calcular raíces cuadradas eficientemente.

El símbolo de Legendre $\eta = \left(\frac{x}{p}\right)$ para un primo p se define como sigue:

$$\left(\frac{x}{p}\right) = \begin{cases} +1 & \text{si } t^2 \equiv \bmod p \text{ es un residuo cuadratico} \\ -1 & \text{si } t^2 \equiv \bmod p \text{ no es un residuo cuadrático} \end{cases}$$

Sean P y Q dos números racionales y α la raíz de $x^2 - Px + Q = 0$ en el campo $\eta(P^2 - 4Q) = -1$ y β el conjugado de α. La función de Lucas $\{U_k\}_{k>0}$ y $\{V_k\}_{k>0}$ con P y Q como parámetros de entrada, está dada por:

$$\begin{aligned} U_k(P, Q) &= \frac{\alpha^k - \beta^k}{\alpha - \beta} \\ V_k(P, Q) &= \alpha^k + \beta^k \end{aligned}$$

donde U_i y V_i satisfacen lo siguiente:

$$\begin{aligned} U_{i+j} &= U_i V_j - Q^j U_{i-j} \\ V_{i+j} &= V_i V_j - Q^j V_{i-j} \end{aligned}$$

En términos de la función de Lucas, la raíz cuadrada u de $Q = \sqrt{x^3 + b}$ módulo p puede ser calculada como sigue.

Sea $k = (p-1)/4$, P un número entero tal que $P^2 - 4Q$ es un residuo no cuadrático. La función de Lucas V_k definida por $V_0 = 2$, $V_1 = P$, $V_k = PV_{k-1} - QV_{k-2}$ entonces u puede ser obtenida calculando $1/2V_{(p+1)/2}$.

Una evaluación eficiente de la función de Lucas puede realizarse con las siguientes formulas:

$$V_{2k} = (V_k)^2 - 2Q^k \tag{2.2}$$

$$V_{2k+1} = V_k V_{k+1} - Q^k P \tag{2.3}$$

de tal manera que la evaluación de $V_{(p+1)/2} = V_{(p+1)/2}(P, Q)$ requiere aproximadamente 4.5 $lg\ r$ multiplicaciones.

La estrategia es conseguir que $Q = 1$ y por tanto las multiplicaciones con Q en las ecuaciones de 2.2 y 2.3 sean omitidas. Así, el parámetro Q en $V_m(Q, P)$ para una m apropiada será igual a 1. Alcanzar tal propiedad es posible con el Algoritmo 2.8, donde $V_m(P, 1)$ puede ser eficientemente calculado con 2 $lg\ m$ multiplicaciones y $lg\ m$ sumas.

Algoritmo 2.7: *Map-to-Point* de Boneh et. al. [12]

Entrada: $msj = \{0,1\}^*$
Salida: $P \in E(\mathbb{F}_p)$

1 $i \leftarrow 0$;
2 Obtener $(x,t) = h'(i||M)$, donde $x \in \mathbb{F}_p$ y $t \in \{0,1\}$;
3 $u = x3 + b$;
4 Calcular la raíz cuadrada $y = u$ sobre $\mathbb{F}_p$;
5 **if** *no se encuentra la solución* **then**
6 $i \leftarrow i + 1$;
7 regresar al paso 2;
8 **end**
9 utilizar t para elegir una de las posibles soluciones y_0 o y_1;
10 **return** $P = (x, y_t)$

Algoritmo 2.8: Evaluación rápida de $V_m(P,1)$ [66]

Entrada: $m = \sum_{j=0}^{l} b_j 2^j$, la representación binaria de m y P.
Salida: $V_m(P,1)$.

1 $d_1 \leftarrow P$; $d_2 \leftarrow P^2 - 2$;
2 **for** $j = l - 1$ ***downto*** 0 **do**
3 **if** $b_j = 1$ **then**
4 $d_1 \leftarrow d_1 d_2 - P$; $d_2 \leftarrow d^2 - 2$;
5 **end**
6 **if** $b_j = 0$ **then**
7 $d_2 \leftarrow d_1 d_2 - P$; $d_1 \leftarrow d^2 - 2$;
8 **end**
9 **end**
10 $w_1 \leftarrow d_1 d_2 - P$; $w_2 \leftarrow d_1^2 - 2$;
11 **if** $b_0 = 1$ **then**
12 **return** w_1
13 **else**
14 **return** w_2
15 **end**

Algoritmo 2.9: Método general para encontrar raíces cuadradas para $p \equiv 1 \bmod 4$ [66]

Entrada: Q.
Salida: Las raíces $\pm u$ de Q en $\mathbb{F}_p$.

```
1  if Q = 4 then
2      return ±2
3  end
4  if η(Q − 4) = −1 then
5      t = 1
6  end
7  if η(Q − 4) = 1 then
8      Elegir t ∈ F_p ;
9      if η(Qt^2 − 4) = 1 then
10         return Error
11     end
12 end
13 P ← Qt^2 − 2;
14 u ← V_{(p−1)/4}(P, 1)/t;
15 return ±u
```

2.6. Sumario

En este capítulo se presentaron las operaciones criptográficas básicas utilizadas por los esquemas de llave pública mostrados a lo largo del este documento.

Cada operación fue considerada de acuerdo al grupo donde se encuentra definido el esquema. Así, se presentaron el fundamento matemático y los algoritmos utilizados para la implementación de las siguientes operaciones:

La exponenciación modular con exponente fijo y con base fija para grupos multiplicativos $\mathbb{Z}_n^*$.

La multiplicación escalar para tres curvas elípticas distintas definidas sobre los campos finitos $\mathbb{F}_p$, $\mathbb{F}_{p^2}$ y $\mathbb{F}_{2^m}$.

La función de emparejamiento bilineal en los grupos $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T)$ definidos como $(\langle P\rangle \in E(\mathbb{F}_p), \langle Q\rangle \in \tilde{E}(\mathbb{F}_{p^2}), \mathbb{F}_{p^k}^*)$ respectivamente.

Por último, la función H_1 cuya funcionalidad es transformar una cadena de longitud arbitraria a un punto en una curva elíptica.

Capítulo 3

Firmas digitales

En este capítulo, se introduce el concepto de firma digital, se enlistan sus propiedades, tipos de ataques y definiciones de seguridad. Además, se presentan los algoritmos y el análisis de seguridad, así como una comparación general y la implementación de cinco esquemas de firma digital considerados en este trabajo como los más importantes publicados en la literatura.

3.1. Definición

En 1976, Diffie y Hellman introdujeron en [26], el concepto de criptografía de llave pública. Los esquemas de llave pública se caracterizan por el hecho de usar un par de llaves, una pública y una privada, ambas asignadas a cada usuario del sistema y relacionadas entre sí. En la criptografía moderna, los esquemas de llave pública son ampliamente usados para generar firmas digitales. La llave privada del usuario, la cual debe ser conocida sólo por su propietario, es requerida por el firmante para producir una única e infalsificable firma digital para un documento dado, mientras que la llave pública debe ser conocida por el verificador de la firma, con la finalidad de decidir si la firma del documento es válida o no.

Notación

- $\mathcal{M}$ representa el conjunto de todos los mensajes que pueden ser firmados.
- $\mathcal{S}$ representa el conjunto de todas las firmas que pueden ser generadas, usualmente con una longitud fija.
- $\mathcal{K}_{\mathcal{S}}$ representa el conjunto de llaves privadas.
- $\mathcal{K}_{\mathcal{V}}$ representa el conjunto de llave públicas.
- $\mathcal{S}_{\mathcal{E}} : \mathcal{M} \times \mathcal{K}_{\mathcal{S}} \rightarrow \mathcal{S}$ representa las reglas de transformación para que la entidad $\mathcal{E}$ produzca una firma.
- $\mathcal{V}_{\mathcal{E}} : \mathcal{M} \times \mathcal{K}_{\mathcal{V}} \rightarrow \{verdadero, falso\}$ representa las reglas de verificación de la transformación para una firma producida por $\mathcal{E}$. Estas reglas son usadas por las

entidades que requieran la verificación de la firma.

Una firma digital se puede definir de la siguiente forma [79]:

Definición 3.1.1. Un esquema de firma digital es una tripleta de algoritmos (*Genera, Firma, Verifica*) tales que,

i. *Genera* es el algoritmo de generación de llaves. Tiene como entrada un parámetro de seguridad ℓ y como salida un par $(k_{\mathcal{S}}, k_{\mathcal{V}}) \in \mathcal{K}_{\mathcal{S}} \times \mathcal{K}_{\mathcal{V}}$, correspondiente a la llave privada y pública, respectivamente.

ii. *Firma* es el algoritmo de firma. Tiene como entrada $(m, k_{\mathcal{S}}) \in \mathcal{M} \times \mathcal{K}_{\mathcal{S}}$ y como salida un elemento $\sigma \in \mathcal{S}$, el cual es la firma del mensaje m producida con la llave privada $k_{\mathcal{S}}$.

iii. *Verifica* es el algoritmo de verificación. Tiene como entrada $(m, \sigma, k_{\mathcal{V}}) \in \mathcal{M} \times \mathcal{S} \times \mathcal{K}_{\mathcal{V}}$ y como salida un valor *verdadero* para indicar una firma válida o un valor *falso* en caso contrario. Se dice que una firma es válida si para cualquier tripleta $(m, \sigma, k_{\mathcal{V}})$ se cumple que

$$Verifica(m, Firma(m, k_{\mathcal{S}}), k_{\mathcal{V}}) = verdadero$$

para cada $(k_{\mathcal{S}}, k_{\mathcal{V}})$ obtenido del algoritmo *Genera* y para cada $m \in \mathcal{M}$.

3.2. Funcionalidad

En las firmas digitales, las funciones picadillo son usadas para producir digestos de mensajes de cualquier longitud. Una función picadillo es una función de sólo ida [47] y se define de la siguiente manera [85]:

Definición 3.2.1. Una *función picadillo* (*hash*) $H : \{0,1\}^* \to \{0,1\}^n$ es una función de sólo ida que convierte una cadena binaria de longitud arbitraria a una cadena binaria de longitud fija. Se dice que $H(x)$ es el digesto de $x \in \{0,1\}^*$, correspondiente a H, de longitud n.

H se considera segura si los siguientes problemas son difíciles de resolver [85]: la *preimagen*, es decir, dado $y \in \{0,1\}^n$, encontrar $x \in \{0,1\}^*$ tal que $H(x) = y$; la *segunda preimagen* que dado un elemento $x \in \{0,1\}^*$ se requiere encontrar un $x' \in \{0,1\}^*$ tal que $x \neq x'$ y $H(x) = H(x')$ y las *colisiones*, esto es, encontrar $x, x' \in \{0,1\}^*$ tal que $H(x) = H(x')$.

Además debe satisfacer dos propiedades muy importantes: ser eficiente y garantizar que un cambio, de incluso 1 bit, a la entrada produzca una salida diferente.

Para efectos de eficiencia, una firma digital es producida para el digesto del mensaje. Las Figuras 3.1 y 3.2 muestran el funcionamiento de un esquema de firma digital. El remitente, que en este documento denominamos *signatario*, usa su llave privada k_S para producir la firma σ. En la verificación, únicamente si la firma fue producida con la llave privada correspondiente a la llave pública con la que se está verificando y el mensaje es el originalmente firmado, el resultado de la verificación será aceptable.

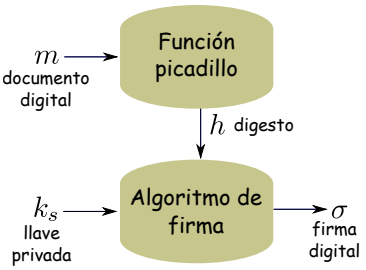

Figura 3.1: Procedimiento de Firma.

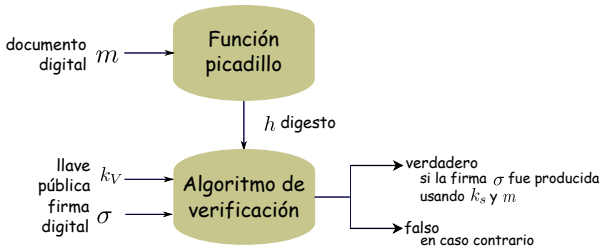

Figura 3.2: Procedimiento de Verificación.

3.3. Propiedades

El concepto de firma digital es análogo al de una firma autógrafa, pero tiene el servicio adicional de proteger la información de alteraciones intencionales de alguna entidad maliciosa. Un esquema de firma digital debe satisfacer las propiedades de integridad, autenticación y no rechazo. La *integridad* implica que el documento recibido sea una copia idéntica del que fue enviado. La *autenticación* asegura que el documento recibido fue creado por una determinada entidad. El *no rechazo* garantiza que tanto el emisor como el receptor no puedan negar haber enviado o recibido algún documento.

Sin embargo, la firma no garantiza por sí misma que la llave pública pertenezca a un determinado usuario. Concretamente, las firmas no pueden garantizar tres casos en particular.

El primero de ellos es una *autenticación segura* entre las entidades denominadas Alicia y Beto, dado que es posible efectuar la usurpación de la identidad donde una tercera entidad denominada Eva establece comunicación con Beto tomando la identidad de Alicia.

El segundo caso es la *revocación de las llaves*, donde la llave privada ha sido comprometida por algún oponente y es necesario que la víctima genere un nuevo par de llaves y que se garantice de algún modo que el viejo e inhabilitado par de llaves no sea usado por alguien más.

El último y tercer caso se refiere al *no rechazo*, el cual es una propiedad de las firmas digitales. El problema radica en que algún usuario puede argumentar que la firma fue

generada con una llave privada que no corresponde con la suya, porque no se utiliza algún mecanismo que avale la relación entre un usuario y su llave pública.

Dado que es posible efectuar los ataques mencionados se creó la Infraestructura de Llave Pública (PKI) donde se usan entidades de confianza denominadas *Autoridades Certificadoras* (AC), las cuales expiden *certificados digitales* que avalan la relación entre un usuario y una llave pública [62]. Un certificado de llave pública es una estructura de datos que consiste en una zona de datos y una zona de firma. La primera incluye la información de identificación del usuario y su llave pública. La segunda contiene la firma digital que certifica la validez de la información en la primera zona, expedida por la Autoridad Certificadora.

La Autoridad Certificadora es la entidad de confianza que certifica la relación entre un usuario y su llave pública, para lo cual requiere un par de llaves propio, donde la llave pública está disponible para todos los usuarios del sistema. Los certificados digitales mantienen una estructura ordenada que permite la rápida localización de la información. En [87] se especifican los campos contenidos en un certificado digital según el estándar X.509 de la Unión Internacional de Telecomunicaciones (ITU) [87].

3.4. Seguridad

La seguridad de los esquemas de firma digital depende de la construcción del par de llaves y de la producción de firmas genuinas. En ambos casos es necesario el uso de *funciones de sólo ida con pasadizo secreto*. Una función de sólo ida f tiene pasadizo secreto si para cualquier argumento x la evaluación $f(x)$ es fácilmente calculada, pero conocida únicamente $f(x)$ es computacionalmente intratable encontrar cualquier $f(y) = f(x)$ sin conocer el *pasadizo secreto*. La Figura 3.3 ilustra el concepto general de este tipo de funciones.

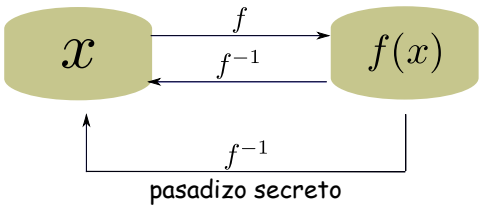

Figura 3.3: Funciones de sólo ida con pasadizo secreto.

Los esquemas de firma digital usan las funciones de sólo ida con pasadizo secreto para garantizar su seguridad. En la producción de la firma, las funciones de sólo ida con pasadizo secreto son utilizadas para evitar que alguna entidad maliciosa produzca una firma falsa sin conocer la llave privada del signatario. Puesto que, bajo ciertos parámetros el problema de resolver f^{-1} sin conocer el pasadizo secreto se convierte en un problema

computacionalmente difícil y por tanto se garantiza que la firma sea infalsificable bajo ese argumento.

La Tabla 3.1 contiene algunos problemas computacionalmente difíciles que son utilizados como supuestos de seguridad en los esquemas de firma digital

Función	Descripción
Factorización de enteros (PFI)	Dado un número entero n, obtener su factorización, es decir, encontrar $n = p_1^{e_1} p_2^{e_2} \cdots p_k^{e_k}$, donde p_i es un número primo y $e_i \geq 0$.
Problema del logaritmo discreto (PLD)	Dado un grupo cíclico $\mathbb{G}$ de orden n, un generador α de $\mathbb{G}$ y un elemento $\beta \in \mathbb{G}$, encontrar un entero x, $0 \leq x \leq n-1$, tal que $\alpha^x = \beta$.
Problema computacional de Diffie-Hellman (PCDH)	Dado un grupo cíclico $\mathbb{G}$ de orden n, un generador α de $\mathbb{G}$ y la tupla (α^a, α^b) con $a, b \in \mathbb{Z}_n$, encontrar α^{ab}.
Problema de decisión de Diffie-Hellman (PDDH)	Dado un grupo cíclico $\mathbb{G}$ de orden n, un generador α de $\mathbb{G}$ y la tupla $(\alpha^a, \alpha^b, \alpha^{ab})$ con $a, b \in \mathbb{Z}_n$, decidir si $ab \equiv c \, mod \, n$.

Tabla 3.1: Problemas computacionalmente difíciles utilizados en los esquemas de firma digital.

3.4.1. Ataques

Un ataque hacia un esquema de firma digital se refiere a posibles acciones por parte de un oponente para romper el esquema. El oponente se denomina comúnmente *adversario* y es un algoritmo de tiempo polinomial que se ejecuta un tiempo determinado. "Romper" el esquema se refiere a obtener información que permita falsificar la firma. Un *modelo de ataque* especifica la información disponible para un adversario cuando se desea romper un esquema. Los algoritmos de firma y verificación se dicen seguros si un adversario no puede alcanzar ciertos objetivos que comprometen la seguridad del esquema completo, aun cuando el adversario tenga en su poder información como la descripción de los algoritmos, las llaves públicas y los mensajes con sus respectivas firmas. En un articulo ya clásico, Goldwasser et al. mencionan en [36] cuatro formas de romper un esquema de firma digital.

i) Rompimiento total: calcular la información secreta de la función con pasadizo secreto de la firma digital.

ii) Falsificación universal: encontrar un algoritmo funcional y equivalente al algoritmo original de firma, basado en información posiblemente diferente pero equivalente a la información secreta de la función con pasadizo secreto en la que se basa la firma

original. En otras palabras, dado un esquema de firma digital $\mathcal{F}$ un adversario $\mathcal{A}$ es un falsificador universal si puede producir una firma falsa para una instancia de $\mathcal{F}$.

iv) Falsificación existencial: falsificar una firma para por lo menos un mensaje. El adversario no tiene el control sobre el mensaje cuya firma obtuvo, por tanto, el mensaje puede ser aleatorio o no tener sentido. En otras palabras, dado un esquema de firma digital $\mathcal{F}$ un adversario $\mathcal{A}$ es un falsificador existencial si para un conjunto de instancias de $\mathcal{F}$ puede producir una firma falsa para por lo menos un mensaje.

En el mismo articulo se definen también dos modelos de ataque. El *ataque de sólo llave*, en el cual el adversario conoce la llave pública del signatario y el *ataque de mensaje*, donde el adversario es capaz de examinar algunas firmas, correspondientes a mensajes cuidadosamente escogidos por él, antes o después de iniciar un ataque. Este último puede dividirse en cuatro tipos, donde $\mathcal{C}$ denota el usuario cuyo esquema de firma está siendo atacado y $\mathcal{A}$ denota al adversario.

i) Ataque del mensaje conocido: $\mathcal{A}$ tiene acceso a las firmas de un conjunto de mensajes $m_1, ..., m_t$. Los mensajes son conocidos por $\mathcal{A}$ pero no son elegidos por él.

ii) Ataque del mensaje escogido genérico: $\mathcal{A}$ tiene permitido obtener de $\mathcal{C}$, firmas válidas para una lista de mensajes escogidos, producidos antes de intentar romper el esquema de firma. Se dice "genérico", debido a que los mensajes son escogidos por $\mathcal{A}$ pero son independientes de la llave pública de $\mathcal{C}$, de tal manera que el ataque puede ser aplicado en contra de cualquier signatario.

iii) Ataque del mensaje escogido dirigido: similar al anterior, con la diferencia de que la lista de mensajes a ser firmados puede ser creada después de conocer la llave pública de $\mathcal{C}$, pero antes de que las firmas sean publicadas. Se dice "dirigido", porque es en contra de un signatario en particular.

iv) Ataque del mensaje escogido adaptativo: $\mathcal{A}$ puede interactuar con $\mathcal{C}$ para solicitar firmas de mensajes que dependen de la llave pública de $\mathcal{C}$ o en su defecto, solicitar firmas de mensajes que dependen de información adicional de las firmas previamente obtenidas. Este ataque es el más poderoso que puede realizarse a los esquemas de firma digital [59].

3.4.2. Demostración de seguridad

Conocidos los ataques y los alcances del adversario, la seguridad de un esquema de firma digital se define considerando el éxito de un adversario *eficiente* para romper el esquema con una probabilidad *desdeñable*. Donde *eficiente* se refiere al tiempo de ejecución del adversario, acotado por un polinomio en función del parámetro de seguridad y *desdeñable*

significa que la probabilidad de que el evento ocurra es más pequeña que el inverso de cualquier polinomio en función del parámetro de seguridad con valores suficientemente grandes [47].

Para demostrar que un esquema es computacionalmente seguro, se necesita demostrar que no existe un adversario eficiente que lo rompa. Dado un esquema de llave pública, la estrategia para demostrar su seguridad es suponer que existe un problema de menor complejidad para resolverse y entonces probar que el esquema en cuestión es seguro bajo esa suposición. Es decir, reducir un algoritmo x que resuelve un problema A en un algoritmo y para resolver un problema B.

En general, la *reducción* convierte un adversario $\mathcal{A}$ que tiene éxito en romper un esquema con una probabilidad no desdeñable a un adversario $\mathcal{A}'$ que tiene éxito en resolver un problema que se supone difícil. Tales problemas pueden ser factorizar un entero grande, resolver el logaritmo discreto, encontrar cierta información en los problemas derivados del intercambio de Diffie-Hellman, entre otros.

Los esquemas de firma digital son demostrados como seguros realizando una o varias reducciones a problemas computacionalmente intratables de tal manera que la seguridad del esquema dependerá de la dificultad de resolver un problema que se presume intratable. Además de la demostración *estándar* de seguridad a través de reducciones, otra forma de realizar una demostración es usando el modelo del *oráculo aleatorio* propuesto por Bellare y Rogaway en [8] donde se modela un ataque como un juego en un espacio de probabilidades entre el *adversario* y el *retador* cuyo esquema de firma está siendo atacado.

Cuando se analiza la seguridad de un esquema de firma bajo el modelo del oráculo aleatorio, el adversario realiza consultas al oráculo a través del retador. En un ataque de mensaje escogido, el adversario puede hacer dos tipos de consultas al oráculo, consultas para picadillos, es decir, que el oráculo simule una función picadillo, y consultas para firmas, lo que significa que el oráculo simulará un algoritmo de firma. Puesto que las interacciones con el oráculo suceden bajo un ataque, las consultas están acotadas en función del parámetro de seguridad establecido por la firma. Es importante resaltar que el objetivo del adversario es producir una firma falsa para algún mensaje ya sea predeterminado o aleatorio, mientras que la finalidad del retador es garantizar que la seguridad del esquema F descansa en la dificultad computacional para invertir alguna función con pasadizo secreto f. Es razonable considerar, entonces, que el retador tenga interés en invertir f y utilice al adversario para conseguir su objetivo. De este modo, la demostración se establece como un juego donde el adversario usa al retador para falsificar una firma y el retador usa al adversario para invertir f. Finalmente, si el adversario consulta correctamente a los oráculos entonces tendrá éxito en falsificar y dado que el retador utiliza al adversario que se supone exitoso entonces el retador tendrá éxito también. De tal manera que el éxito del adversario es el éxito del retador, concluyendo así que el esquema es seguro bajo la suposición de que f es intratable.

3.5. Esquemas de firma digital propuestos

La firma digital propuesta por Rivest et al. en [77], llamada RSA, fue la primera realización práctica de un esquema de firma digital y hasta la actualidad sigue siendo el esquema más utilizado. Se encuentra también, la firma digital ElGamal, propuesta por Taher ElGamal en 1985 [28]. Una versión posterior a esta firma fue denominada DSA y fue estandarizada por el Instituto Nacional de Estándares y Tecnologías (NIST) como DSS en 1994 [67]. A través de los años, varios esquemas de firma digital han sido propuestos. Un ejemplo notorio es la criptografía de curvas elípticas que fue independientemente propuesta por Neal Koblitz [49] y Victor Miller [63] en 1985.

En el marco de la criptografía de curvas elípticas, el esquema más sobresaliente de firma digital es ECDSA [84]. Debido a la complejidad de calcular el problema del logaritmo discreto en curvas elípticas definidas sobre un campo finito, podemos obtener con la firma ECDSA, la misma seguridad provista por otros esquemas de firma digital como RSA o DSA, pero usando llaves de longitud más pequeña.

En 1994, Menezes et al. [61] y Frey y Rück [30] introdujeron en la criptografía las funciones de emparejamiento bilineal de Weil y de Tate, como una herramienta de ataque en contra del problema de logaritmo discreto en algunas curvas definidas sobre campos finitos. Sin embargo, los trabajos de Sakai et al. [80] y Joux [44] mostraron que era posible utilizar los emparejamientos para obtener soluciones a problemas criptográficos de manera constructiva. Desde entonces, la denominada criptografía basada en emparejamientos ha tenido una gran demanda, consiguiendo un incremento en el número de protocolos propuestos año con año. En 2001, Boneh et al. [13] propusieron el primer esquema de firma digital usando emparejamientos y lo denominaron *firma corta*. En 2002, Hess [39] propuso el primer esquema de firma digital que combinaba los emparejamientos con la criptografía basada en la identidad.

A continuación se presentan cinco esquemas de firma digital arriba mencionados, sus algoritmos y algunos comentarios sobre su seguridad.

3.5.1. Firma digital RSA

El esquema de firma digital RSA, es el esquema más utilizado en la actualidad. La generación de llaves, como se puede observar en el Algoritmo 3.1, requiere del parámetro de seguridad ℓ como dato de entrada y tiene como salida la tripleta (d, e, N). Primero se eligen un par de números primos grandes p y q y se calcula el producto $N = pq$, donde N es el módulo RSA. Se elige también un número entero e pequeño, el cual debe ser primo relativo de $\phi = (p-q)(q-1)$. Por último se obtiene $d \equiv e^{-1} \bmod \phi$. La llave privada es (d, N) y la llave pública es (e, N).

En el algoritmo de firma 3.2, el signatario utiliza su llave privada para producir la firma para el mensaje m. El proceso requiere de la función picadillo $H : \{0,1\}^* \rightarrow \{0, N-1\}$ para obtener el digesto del mensaje y producir la firma para el mismo, con

lo cual se garantiza la integridad del mensaje.

El algoritmo de verificación, mostrado en 3.3, toma como entrada el mensaje, la firma y la llave pública del signatario. La entidad verificadora calcula el digesto del mensaje $h = H(m)$ y finalmente verifica que la ecuación $h \equiv s^e \bmod N$ se cumpla. Si es el caso entonces la firma es válida, de lo contrario es rechazada.

Algoritmo 3.1: Generación de llaves RSA [77]

Entrada: Parámetro de seguridad ℓ.
Salida: Llaves (d, e, N).

1 Elegir aleatoriamente dos números primos p y q, de $\ell/2$ bits de longitud;
2 $N = pq$;
3 $\phi(N) = (p-1)(q-1)$, con $mcd(e, \phi(N)) = 1$;
4 Elegir aleatoriamente un número pequeño e;
5 d tal que $d \equiv e^{-1} \bmod \phi(N))$;
6 **return** (d, e, N)

Algoritmo 3.2: Firma RSA [77]

Entrada: Llave privada (d, N), mensaje m.
Salida: Firma s.

1 $h = H(m)$;
2 $s \equiv h^d \bmod N$;
3 **return** s

Algoritmo 3.3: Verificación RSA [77]

Entrada: Llave pública (e, N), mensaje m, firma s.
Salida: {Válida/Rechazada}.

1 $h = H(m)$;
2 $h' \equiv s^e \bmod N$;
3 **if** $h = h'$ **then**
4 **return** *Válida*;
5 **else**
6 **return** *Rechazada*;
7 **end**

Respecto a la seguridad, el esquema puede ser atacado de diferentes maneras, a continuación se enlistan algunas de ellas.

Falsificación existencial bajo el ataque de mensaje escogido. El adversario tiene dos firmas con sus respectivos mensajes, producidas de forma válida por el signatario. Entonces, el adversario produce un tercer mensaje calculando,

$$s_3 \equiv s_1 s_2 \equiv (m_1)^d (m_2)^d \equiv (m_1 m_2)^d \bmod N$$

la firma s_3 con el nuevo mensaje $m_3 = m_1 m_2$ será valido ya que la siguiente ecuación se cumple.

$$s_3^e \equiv (m_1^d m_2^d)^e \equiv (m_1 m_2)^{ed} \bmod N \equiv m_3$$

Solución. Con el uso de la función picadillo, la generación del tercer mensaje implicaría obtener una preimagen, es decir, dado $h_1 = H(m_1)$, $h_2 = H(m_2)$ y $h_3 = h_1 h_2$ obtener el mensaje m_3 del picadillo h_3. Si la función picadillo es segura en contra de la preimagen entonces para el adversario será computacionalmente difícil producir una firma falsa.

Rompimiento total factorizando el módulo. El adversario puede usar los algoritmos más eficientes para factorizar el módulo N, si los números primos no son seleccionados de forma correcta o con suficiente longitud en bits.

Solución. La longitud de p y q deben ser de un tamaño suficiente para que la factorización de su producto sea un problema intratable computacionalmente. Así, el módulo N es el producto de los primos p y q tales que: $|p| = |q| = 512$-bits de longitud como mínimo, si se quiere conseguir una seguridad de por lo menos 80-bits. Además de la longitud en bits, también es necesario considerar algunas restricciones en la elección de p y q como que es preferible que sean primos fuertes que primos generados aleatoriamente. Un primo p es fuerte si dados los enteros r, s y t se cumple que $p-1$, $p+1$ y $r-1$ tienen como factor un primo largo r, s y t respectivamente. Tales condiciones permiten que los primos fuertes ofrezcan más de protección que los primos aleatorios [62].

3.5.2. Firma digital DSA

En 1991, el Instituto Nacional de Estándares y Tecnologías de Estados Unidos (NIST) propuso un esquema de firma digital denominado *Digital Signature Algorithm* (DSA), el cual fue estandarizado en 1994 como Digital Signature Standard (DSS) [67], convirtiéndose en el primer esquema de firma digital reconocido ante un gobierno. DSA es una variación de la firma digital ElGamal y requiere de una función picadillo de la forma $H : \{0,1\}^* \to \mathbb{Z}_q$, para algún entero q. Según el estándar FIPS-186, la firma digital usa específicamente la función picadillo SHA-1 [68].

La generación de parámetros se muestra en el Algoritmo 3.4. Primero son seleccionados dos primos q y p tales que q divide a $p-1$, lo que implica que $(\mathbb{Z}/p\mathbb{Z})^*$ contiene elementos de orden q. Después, se selecciona un elemento $g \in \mathbb{F}_p^*$ y se obtiene un generador $\alpha = g^{(p-1)/q} \bmod p$ de orden q en $(\mathbb{Z}/p\mathbb{Z})^*$. Finalmente, se elige un elemento $a \in \mathbb{F}_q^*$

y se calcula $y \equiv \alpha^a \bmod p$. La llave privada es a y la llave pública es la tupla (p, q, α, y). Calcular a a partir de y requiere resolver el problema del logaritmo discreto en $\mathbb{Z}_p^*$.

El Algoritmo 3.5 muestra el procedimiento para obtener una firma. Para firmar el mensaje m, el signatario obtiene el digesto de m usando una función picadillo de la forma $H : \{0,1\}^* \to \{1, 2, ...q-1\}$; elige un número $k \in \mathbb{Z}_q^*$ denominado en este documento *llave de sesión*; calcula $r \equiv (\alpha^k \bmod p) \bmod q$ y $s \equiv k^{-1}(H(m) + ar) \bmod q$. La firma del mensaje m es el par (r, s). Cualquier entidad puede verificar la autenticidad de la firma utilizando el algoritmo 3.6. Si la igualdad $\alpha^k \equiv \alpha^{s^{-1}H(m)}\alpha^{as^{-1}r} \bmod p$ se cumple, la firma es válida de otra manera es rechazada. Algunos de los ataques más emblemáticos en contra de la firma digital DSA son los siguientes:

Rompimiento total usando la llave de sesión. El signatario usa la misma llave de sesión k para producir dos firmas s_1, s_2 de dos mensajes distintos m_1, m_2,

$$r_1 = r_2 \equiv (\alpha^k \bmod p) \bmod q \text{ y } s_1 - s_2 \equiv k^{-1}(H(m_1) - H(m_2)) \bmod q$$

con lo cual, se obtiene la llave de sesión, calculando,

$$k \equiv (s_1 - s_2)^{-1}(H(m_1) - H(m_2)) \bmod q$$

con la llave de sesión al descubierto, la llave privada puede obtenerse de la siguiente manera,

$$a \equiv r^{-1}(H(m_1) - ks_1) \bmod q$$

Solución. La llave de sesión debe ser única por cada firma producida por el signatario.

Rompimiento total resolviendo el PLD. El adversario puede usar los algoritmos más eficientes para resolver el PLD en $\mathbb{F}_p^*$, si los números primos no son seleccionados de forma correcta o con suficiente longitud en bits.

Solución. Dada la eficiencia de los algoritmos para resolver el PLD, los primos p y q deben ser seleccionados de tal manera que los algoritmos sean imprácticos. Para conseguir una seguridad de 80 bits, la longitud de p debe ser un múltiplo de 64 entre 512 y 1024 bits. Con estas condiciones, el algoritmo más eficiente para resolver el PLD en un grupo de orden q es el de Pollard ρ, el cual requiere calcular más de $\sqrt{q}$ operaciones, dado que $q > 2^{159}$ y $p < 2^{1024}$ es impráctico resolver el PLD con la tecnología actual.

Algoritmo 3.4: Generación de llaves DSA [67]

Entrada: Parámetro de seguridad ℓ.
Salida: Llave privada: a, llave pública: (p, q, α, y).

1 Seleccionar un primo q tal que $2^{255} < q < 2^{256}$ y $q|p-1$;
2 Seleccionar un primo p tal que $2^{\ell-1} < p < 2^{\ell}$;
3 Elegir $g \in \mathbb{F}_p^*$ y calcular $\alpha \equiv g^{(p-1)/q} \bmod p$;
4 Seleccionar aleatoriamente un número a tal que $1 \leq a \leq q-1$;
5 $y \equiv \alpha^a \bmod p$;
6 **return** (a, p, q, α, y)

Algoritmo 3.5: Firma DSA [67]

Entrada: Llave privada a, el mensaje m.
Salida: Firma (r, s).

1 $h = H(m)$;
2 Seleccionar aleatoriamente $k \in \mathbb{Z}_q$;
3 $r \equiv \alpha^k \bmod q$;
4 $s \equiv k^{-1}(h + ar) \bmod q$;
5 **return** (r, s)

Algoritmo 3.6: Verificación DSA [67]

Entrada: llave pública (p, q, α, y), el mensaje m y la firma (r, s).
Salida: {Válida/Rechazada}.

1 $h = H(m)$;
2 $w \equiv s^{-1} \bmod q$;
3 $u_1 \equiv hw \bmod q$;
4 $u_2 \equiv rw \bmod q$;
5 $v \equiv (\alpha^{u_1} y^{u_2} \bmod p) \bmod q$;
6 **if** $r = v$ **then**
7 **return** *Válida*;
8 **else**
9 **return** *Rechazada*;
10 **end**

3.5.3. Firma digital ECDSA

La firma digital ECDSA es una variación de la firma digital DSA que utiliza puntos definidos sobre una curva elíptica. Es un algoritmo propuesto por el Instituto Nacional de Normas y Tecnología de los Estados Unidos para su uso en su estándar especificado

en el FIPS 186. Los parámetros de dominio necesarios para obtener un criptosistema basado en el problema del logaritmo discreto para curvas elípticas (PLDCE) son los siguientes. Una curva elíptica E definida sobre un campo primo $\mathbb{F}_q$ con $q = p^m$, p primo y $m \geq 1$ y un punto generador $P \in E(\mathbb{F}_q)$ de orden n.

En la generación de llaves, mostrada en el Algoritmo 3.7, el signatario selecciona su llave privada d en el intervalo $[2, n-1]$ y obtiene la llave pública calculando la multiplicación escalar $Q = dP$. Por tanto, la llave privada d está protegida con el PLDCE, dado que, obtener d a partir de V es computacionalmente impráctico.

Para producir una firma, como se muestra en el Algoritmo 3.8, el procedimiento es muy similar al del algoritmo de firma DSA. El signatario selecciona una llave de sesión k y calcula la multiplicación escalar $R = kP$. r es la coordenada x del punto R convertido en un entero módulo n. s es calculado de la misma forma que en DSA, utilizando r, d y el digesto de m obtenido de la función picadillo, ahora de la forma $H : \{0,1\}^* \rightarrow \{1, 2, ..., n-1\}$. La firma producida por ECDSA es el par (r, s) ambos módulo n.

En la verificación de la firma, indicada en el Algoritmo 3.9, si la ecuación $s \equiv k^{-1}(h + dr)$ es válida, implica que $k \equiv s^{-1}h + s^{-1}dr \equiv wh + wdr \equiv u_1 + u_2 d$, sustituyendo valores según las líneas 3 y 4 del Alg. 3.9. Si $kP = u_1 P + u_2 Q$ se cumple entonces la firma es válida, de otra manera es rechazada. Los ataques que un esquema de firma digital ECDSA puede tener son los siguientes.

Falsificación existencial bajo el ataque de mensaje escogido. Si la función picadillo no es resistente a la preimagen, entonces el adversario puede generar una firma falsa de la siguiente manera. r es la coordenada x del punto $Q + lP$, donde l es un entero en $\mathbb{Z}_n$, después asigna $s = r$ y calcula $e = rl \bmod n$. Si el adversario puede encontrar un mensaje m que satisfaga $e = H(m)$, entonces (r, s) será una firma válida para m.

Rompimiento total usando la llave de sesión. A pesar de que el PLDCE es más difícil de resolver que el PLD en enteros, la firma digital ECDSA tiene la misma debilidad que la firma DSA, al usar la misma llave de sesión. Lo anterior, es debido a que el cálculo de s es con operaciones sobre enteros y no con puntos sobre la curva elíptica, por tanto, puede resolverse la ecuación $k \equiv (s_1 - s_2)^{-1}(H(m_1) - H(m_2)) \bmod n$ para dos firmas producidas con la misma llave de sesión.
Solución. Como en DSA, es utilizar por única ocasión a k para cada firma producida.

Rompimiento total resolviendo el PLDCE. El adversario puede usar los algoritmos más eficientes para resolver el PLD sobre curvas elípticas, si la curva elíptica no es correctamente definida sobre un campo finito adecuado.

Solución. Para conseguir una seguridad de por lo menos 80 bits, es decir, que se requieran de por lo menos 2^{80} operaciones para resolver el PLDCE, la curva elíptica debe ser

definida sobre un campo primo finito en el rango de 160 a 233 bits. El NIST recomienda una lista de curvas que contemplan cinco niveles de seguridad de 80, 112, 128, 192 y 196 bits.

Algoritmo 3.7: Generación de llaves ECDSA [84]

Entrada: La curva elíptica E, punto generador P, orden n.
Salida: Llave privada d, llave pública Q.

1 Seleccionar aleatoriamente $d \in \mathbb{Z}_n$;
2 $Q = dP$;
3 **return** (d, Q)

Algoritmo 3.8: Firma ECDSA [84]

Entrada: La llave privada d, el mensaje m.
Salida: Firma (r, s).

1 $h = H(m)$;
2 Seleccionar aleatoriamente $k \in \mathbb{Z}_n$;
3 $kP = (x_1, y_1)$;
4 $r \equiv x_1 \bmod n$;
5 $s \equiv k^{-1}(h + dr) \bmod n$;
6 **return** (r, s)

Algoritmo 3.9: Verificación ECDSA [84]

Entrada: La llave pública Q, el mensaje m y la firma (r, s).
Salida: {Válida/Rechazada}.

1 $h = H(m)$;
2 $w \equiv s^{-1} \bmod n$;
3 $u_1 \equiv hw \bmod n$;
4 $u_2 \equiv rw \bmod n$;
5 $X = (x_X, y_X) = u_1 P + u_2 Q$;
6 $v \equiv x_X \bmod n$;
7 **if** $r = v$ **then**
8 **return** *Válida*;
9 **else**
10 **return** *Rechazada*;
11 **end**

3.5.4. Firma corta

En 2001, Boneh et al. [13] propusieron el primer esquema de firma basada en emparejamientos denominada *firma corta*. La firma corta como su nombre lo indica produce una firma más pequeña que las producidas por RSA y DSA, lo cual la hace útil en ambientes con limitaciones ancho de banda.

Los parámetros de dominio de la firma corta son los grupos aditivos $\mathbb{G}_1$, $\mathbb{G}_2$, el grupo multiplicativo $\mathbb{G}_T$, el primo r, un punto $P \in \mathbb{G}_1$ con $|\mathbb{G}_1| = |\mathbb{G}_2| = |\mathbb{G}_T| = r$ y además utiliza la función *map-to-point* y la función de emparejamiento $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$.

Una característica de los esquemas basados en emparejamientos es que el problema computacional de Diffie-Hellman es computacionalmente difícil de resolver mientras que el problema de decisión de Diffie-Hellman es fácil en los grupos $\mathbb{G}_1$ y $\mathbb{G}_2$. Específicamente, la seguridad de la firma corta descansa en los problemas computacionales PCDH y PLDCE (véase Tabla 3.1).

El Algoritmo 3.10 muestra la generación de las llaves. La llave privada es seleccionada en el intervalo $[2, r-1]$ y la llave pública se calcula realizando una multiplicación escalar, por tanto, d está protegida por el PLD en curvas elípticas.

El Algoritmo 3.11 indica los pasos para producir una firma. Primero el signatario convierte el mensaje a un punto sobre la curva elíptica utilizando H_1, después con una simple multiplicación escalar en el grupo $\mathbb{G}_1$ produce la firma. A diferencia de los esquemas anteriores, la firma producida es un punto sobre la curva elíptica.

El Algoritmo 3.12, es el algoritmo de verificación para la firma corta, la cual comprende de la comparación de los resultados obtenidos después del cálculo de dos emparejamientos. La firma es válida si la siguiente relación se cumple.

$$\begin{aligned} \hat{e}(P, S) &= \hat{e}(P, dH_1(m)) \\ &= \hat{e}(P, H_1(m))^d \\ &= \hat{e}(dP, H_1(m)) \\ &= \hat{e}(Q, H_1(m)) \end{aligned}$$

De lo contrario es rechazada. Nótese que los puntos $(P, S, Q, H_1(m))$ forman una tupla donde puede definirse el PDDH y la función de emparejamiento es útil para decidir si S es equivalente a $dH_1(m)$, dados los puntos Q, V y $H_1(m)$. Los ataques contemplados para este tipo de firmas son los siguientes.

Falsificación existencial bajo el ataque de mensaje escogido. Si la firma corta es producida sin utilizar la función H_1. Sea un mensaje m y el punto base P, el adversario utiliza la llave pública del signatario $Q = dP$ para producir la firma falsa $S_f = mQ =$

$mdP = dmP$. La verificación de la firma falsa es como sigue.

$$\begin{aligned} \hat{e}(P,S) &= \hat{e}(P,dmP) \\ &= \hat{e}(P,mP)^d \\ &= \hat{e}(dP,mP) \\ &= \hat{e}(Q,mP) \end{aligned}$$

Solución. Si la función H_1 es resistente a la preimagen entonces para el adversario es un problema intratable falsificar una firma debido a la dificultad de encontrar un punto mP tal que $mP = H_1(m)$.

Rompimiento total resolviendo el PLDCE. La generación de las llaves y la producción de la firma se realiza calculando multiplicaciones escalares en los grupos aditivos. Por tanto, la curva elíptica debe ser definida de tal manera que el PLDCE sea intratable en esos grupos. En la verificiación, dado que los puntos S y Q son conocidos y M es obtenido con la función H_1, es importante también que en los grupos sea intratable el PCDH y a la vez pueda resolverse el PDDH.

Solución. Para alcanzar un nivel de seguridad de 128 bits, la firma corta requiere ser definida sobre grupos con un orden especifico para garantizar que el PLDCE y PCDH sean problemas computacionalmente intratables en los grupos $\mathbb{G}_1, \mathbb{G}_2$.

Boneh et al. demostraron que la firma corta es segura a través de una reducción al problema denominado *DHP1* definido como sigue:

Sea $\mathbb{G}$ un grupo de orden primo n, sea g un elemento generador de $\mathbb{G}$. Dados $X = g^x \bmod n$ y $Y = g^y \bmod n$, existe un oráculo $\mathcal{O}^H$ que tiene como salida puntos aleatorios en $\mathbb{G}$ y otro oráculo $\mathcal{O}^{CDH}$ que dados (X, Y) resuelve el problema computacional de Diffie-Hellman (PCDH), es decir, dados $(X = g^x, Y = g^y)$, el DHP1 es el problema de calcular $Z = g^{xy}$, con la restricción de que el número de consultas a $\mathcal{O}^{PCDH}$ es estrictamente menor al número de consultas a $\mathcal{O}^H$.

El adversario $\mathcal{A}$ puede realizar q_H y q_{CDH} consultas a los oráculos $\mathcal{O}^H$ y $\mathcal{O}^{CDH}$, respectivamente. Las consultas a $\mathcal{O}^{CDH}$ dan como resultado un conjunto de pares (Z_i, z_i) generados aleatoriamente, los cuales son indistinguibles del conjunto de pares que genera el adversario $\mathcal{A}$ comenzando con un z_i y calculando $Z_i = g^{z_i}$. Si el adversario fija X y calcula $Z_i = X^{y_i}$ para un y_i aleatorio, entonces el PDH1 se reduce al PCDH en $\mathbb{G}_1$ para una instancia (X, Y_i) del PCDH.

La definición de seguridad para el esquema de firma corta de Boneh et al. es como sigue:

Definición 3.5.1. El esquema de firma corta es seguro contra un falsificador existencial $\mathcal{A}$ utilizando el ataque de mensaje escogido bajo el modelo del oráculo aleatorio, si el DHP1 es seguro.

La demostración se desarrolla como un juego entre el falsificador $\mathcal{A}$ y el adversario $\mathcal{B}$ que manipula los oráculos aleatorios. Ambas entidades son algoritmos de tiempo polinomial. El juego comienza con las consultas hechas por $\mathcal{A}$ primero a $\mathcal{O}^H$. Después de tener varios puntos aleatorios, $\mathcal{A}$ consulta el oráculo de firma, para lo cual $\mathcal{B}$ consulta a $\mathcal{O}^{DHP}$. Al final de las consultas $\mathcal{A}$ obtiene una lista de tuplas $(m_1, M_1, S_1), ..., (m_\ell, M_\ell, S_\ell)$, mientras que $\mathcal{B}$ para cada consulta hecha por $\mathcal{A}$ fija M y calcula $S_i = M^{y_i}$ para un y_i aleatorio que satisfaga la verificación de la firma. Evidentemente, $\mathcal{B}$ tendrá éxito si $\mathcal{A}$ tiene éxito, de tal manera que la falsificación se produce si para una instancia de la firma, el $\mathcal{B}$ puede resolver el DHP1. Lo que significa resolver para una instancia del PCDH, por tanto se concluye que el esquema de firma corta es seguro en contra del ataque del mensaje escogido adaptativo.

Algoritmo 3.10: Generación de llaves Firma corta [13]

Entrada: La curva elíptica E, punto generador P de orden r.
Salida: Llave privada: d, llave pública: Q.

1 Seleccionar aleatoriamente $d \in \mathbb{Z}_r$;
2 $Q = dP$;
3 **return** (d, Q)

Algoritmo 3.11: Firma Corta [13]

Entrada: El mensaje m, la llave privada d.
Salida: Firma S.

1 $M = H_1(m)$;
2 $S = dM$;
3 **return** S

Algoritmo 3.12: Verificación Firma Corta [13]

Entrada: El mensaje m, la firma S y la llave pública Q.
Salida: {Válida/Rechazada}.

1 $M = H_1(m)$;
2 **if** $\hat{e}(P, S) = \hat{e}(Q, M)$ **then**
3 **return** *Válida*;
4 **else**
5 **return** *Rechazada*;
6 **end**

3.5.5. Firma digital basada en la identidad

Preliminares

Los cripto-sistemas basados en la identidad (IBC, por sus siglas en inglés) son sistemas de llave pública que requieren una transformación privada y otra pública, con la distinción de que la llave pública puede ser cualquier información del usuario que lo identifique de forma única. Este concepto fue introducido por Shamir en 1984 [82] y en [62] lo definen como un sistema de llave pública donde la información pública que identifica a un usuario de forma única, juega el rol de llave pública y es usada por una autoridad de confianza para calcular la correspondiente llave privada.

La autoridad de confianza denominada *Generador de Llaves Privadas* (PKG, por sus siglas en inglés) requiere de un par de llaves propio, donde la llave pública es utilizada en conjunto con la llave pública del usuario para producir la llave privada del mismo. Una vez obtenida la llave privada, el PKG, la envía al usuario usando un canal privado. La motivación principal de este tipo de sistemas es eliminar la necesidad de transmitir las llaves públicas y permitir a los protocolos utilizar información pública única para autenticar la llave pública del usuario. Esta particularidad ofrece una ventaja a los algoritmos de firma digital, donde el certificado de la llave pública no es necesario dado que la información pública del usuario es suficiente para identificarlo.

La Figura 3.4 muestra un esquema de firma digital usando criptografía basada en la identidad.

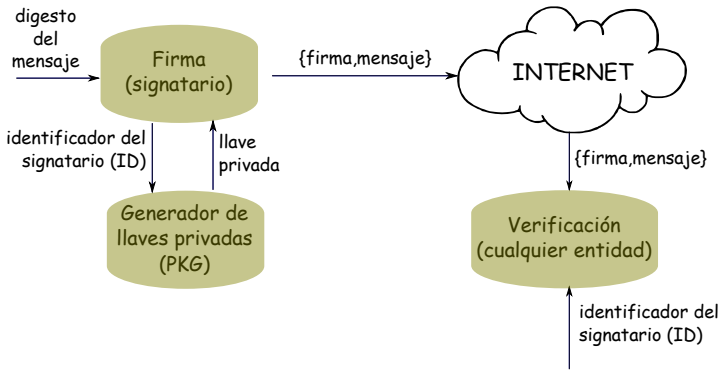

Figura 3.4: Firma y verificación basados en la identidad

En el contexto de la criptografía basada en la identidad, se han propuesto esquemas de firma utilizando emparejamientos. A diferencia de la firma corta, este tipo de esquemas requieren de la generación de un par de llaves para la autoridad de confianza denominada *Generador de Llaves*, encargada de generar las llaves de todos los usuarios. El Algoritmo 3.13 muestra el procedimiento.

El primer esquema de firma que combina los emparejamientos con la criptografía basada en la identidad fue el propuesto por Hess en el 2002 [39]. Como la firma corta,

el esquema de Hess basa su seguridad en los PCDH y PLDCE (véase Tabla 3.1).

El Algoritmo 3.14 muestra la generación de llaves para el usuario, donde se puede observar que la llave pública es cualquier información pública que identifica al usuario de forma única, mientras que la llave privada se convierte de entero a un punto en $\mathbb{G}_1$.

Los Algoritmos 3.15 y 3.16 indican los pasos a seguir para obtener y verificar una firma. La firma producida consta del par $(U, v) \in \mathbb{G}_1 \times \mathbb{Z}_r^*$. Usa una función especial de la forma $H : \{0,1\}^* \times \mathbb{G}_T \to \mathbb{Z}_r^*$. En la verificación la firma es válida si la igualdad $\hat{e}(Q, U) = \hat{e}(-P_{pub}, Q_{ID})$ se satisface.

Algoritmo 3.13: Generación de llaves para el PKG [39]

Entrada: La curva elíptica E, punto generador P, orden r.
Salida: Llave privada: s, llave pública: P_{pub}.

1 Seleccionar aleatoriamente $s \in \mathbb{Z}_r$;
2 $P_{pub} = sP$;
3 **return** (s, P_{pub})

Algoritmo 3.14: Generación de llaves [39]

Entrada: Llave privada del PKG s, ID del signatario.
Salida: Llave privada del signatario: S_{ID}.

1 $Q_{ID} = H_1(ID)$;
2 $S_{ID} = dQ_{ID}$;
3 **return** (S_{ID})

Algoritmo 3.15: Firma digital basada en la identidad [39]

Entrada: La llave privada S_{ID}.
Salida: Firma (U, v).

1 Seleccionar aleatoriamente $k \in \mathbb{Z}_r$;
2 Seleccionar aleatoriamente $P_1 \in \mathbb{G}_1$;
3 $r_1 = \hat{e}(P, P_1)^k \in \mathbb{G}_T$;
4 $v = H(m, r_1)$;
5 $U = vS_{ID} + kP_1 \in \mathbb{G}_1$;
6 **return** (v, U)

Algoritmo 3.16: Verificación Firma digital basada en la identidad [39]

Entrada: La firma (u, v), el mensaje m y el ID del signatario.
Salida: {Válida/Rechazada}.

```
1 if ê(P, U) = ê(−P_pub, H_1(ID)) then
2     return Válida;
3 else
4     return Rechazada;
5 end
```

Cabe mencionar que la utilidad de los emparejamientos bilineales en los esquemas criptográficos y tomando en cuenta el concepto de la criptografía basada en la identidad (IBC). En el artículo denominado "Achieving Identity-Based Cryptography in a Personal Digital Assistant Device"[60], Martínez et al. propusieron una aplicación criptográfica que permite el intercambio seguro de documentos de un Asistente Digital Personal (PDA), el cual es conectado de forma inalámbrica con otros dispositivos. La aplicación está inspirada en el tradicional protocolo de seguridad de correo electrónico PGP (Pretty Good Privacy) [14], y tiene los servicios de autenticación y confidencialidad utilizando esquemas de cifrado y firma digital basados en la identidad.

3.6. Comparación general

Las firmas digitales pueden ser definidas en grupos aditivos o multiplicativos, como puede observarse en la Tabla 3.2. En el primer renglón se encuentran las firmas RSA y DSA las cuales operan sobre números enteros módulo N en el primer caso y módulo p o q para el segundo. Sigue el esquema ECDSA que utiliza la operación aditiva con puntos sobre una curva elíptica. Por último, se consideran la firma corta y la firma IBE las cuales utilizan la transformación de grupos aditivos a grupos multiplicativos a través de la función de emparejamiento bilineal.

Enteros	Puntos sobre una curva elíptica	Emparejamientos	
RSA, DSA	ECDSA	Firma corta	Firma IBE

Tabla 3.2: Algoritmos de firma digital.

Dado que los esquemas mencionados en la sección anterior están definidos en distintos grupos, para realizar la comparación de seguridad entre los esquemas es necesario establecer el *nivel de seguridad*, el cual es el número mínimo de operaciones requeridas para romper la seguridad de un esquema. Actualmente el mínimo de operaciones para considerar un nivel de seguridad estable pero no altamente seguro es de 2^{80} operaciones. En esta tesis, es considerado un nivel 128 bits, es decir, se deben de realizar como mínimo

2^{128} operaciones para romper cualquiera de los esquemas presentados en los capítulos siguientes.

La Tabla 3.3 muestra la longitud en bits de los parámetros que son definidos en cada esquema de firma digital para brindar un nivel de seguridad de 128 bits.

RSA produce una firma con una longitud de 3072 bits. DSA obtiene una firma con una longitud de 256 bits para cada uno de sus elementos y en la versión en curvas elípticas es de 251 bits.

Para los esquemas basados en emparejamientos, ambos pueden utilizar emparejamientos simétricos y asimétricos. Sin embargo, en este documento únicamente se consideran los emparejamientos asimétricos por ser más eficientes que los simétricos.

La función de emparejamiento asimétrico utilizada es el *ate óptimo* (ver definición 2.4.3), el cual está definido para una curva ordinaria sobre el campo finito primo $\mathbb{F}_p$ donde $|p| = 254$ bits con orden $|r| = 254$ bits. Entonces, la firma corta tiene una longitud 254 bits, mientras que para la firma de Hess es de 508 bits.

Esquemas	Parámetros	Supuesto de seguridad	Longitud de las firmas
RSA	$\|N\| = 3072$ bits	PF	3072-bits
DSA	$\|p\| = 3072$ bits $\|q\| = 256$ bits	PLD	$(256, 256)$-bits
ECDSA	$\|n\| = 251$ bits	PLDCE	$(251, 251)$-bits
Firma corta	$\|r\| = 254$ bits	PCDH	$(254, 1)$-bits
Firma IBE	$\|r\| = 254$ bits	PCDH	$((254, 1), 254)$-bits

Tabla 3.3: Comparación de los esquemas de firma digital ofreciendo un grado de seguridad de 128 bits.

3.7. Comparación de eficiencia

Para comparar el desempeño, se implementaron todos los esquemas de firma digital con los parámetros de dominio definidos para ofrecer un nivel de seguridad de 128 bits.

Las operaciones privada y pública RSA son exponenciaciones módulo N con exponentes de aproximadamente 3072 y 17 bits respectivamente. La exponenciación para los esquemas basados en el problema de logaritmo discreto módulo p es menos costosa que la operación privada de RSA debido a que el exponente es del orden de q el cual tiene 256 bits de longitud.

Para el esquema basado en la criptografía sobre curvas elípticas, la operación principal es la multiplicación escalar sobre un grupo aditivo generado por un punto base P perteneciente a la curva elíptica ordinaria E definida sobre un campo primo binario $\mathbb{F}_{2^{251}}$ de orden $|n| = 251$ bits.

En el contexto de la criptografía basada en emparejamientos, la función de emparejamiento está definida para los grupos $\mathbb{G}_1$, $\mathbb{G}_2$ y $\mathbb{G}_T = \langle g \rangle \in \mathbb{F}^*_{p^{12}}$ son de orden r, con $|r|$=254 bits.

La Tabla 3.4 muestra el número de ciclos que requiere cada esquema para producir y verificar una firma digital con una seguridad de 128 bits. La implementación de los esquemas se desarrolló utilizando varias bibliotecas. Para RSA y DSA usamos la biblioteca MIRACL (Multiprecision Integer and Rational Arithmetic C/C++ Library) [81], la cual contiene todas las primitivas necesarias para implementar los esquemas de firma digital RSA y DSA. La multiplicación escalar para la firma ECDSA se basa en la implementación realizada por Taverne et al. [86], la cual utiliza el método de bisección y suma de puntos. Por último, las operaciones criptográficas para emparejamientos asimétricos fueron diseñadas basadas en la implementación de Beuchat et al. [9] sobre curvas BN combinadas con el soporte criptográfico de la biblioteca MIRACL. El código fue ejecutado en un equipo de cómputo con un procesador Intel Core 2 Quad @2.4Ghz y los algoritmos utilizados para cada operación criptográfica se muestran en el Capítulo 2.

Es notorio que la firma ECDSA es la más eficiente seguida de la firma corta de Boneh, en contraste con los esquemas DSA y RSA que son altamente costosos.

3.8. Sumario

A lo largo de este capítulo se presentaron cinco esquemas de firma digital, definidos sobre grupos tanto aditivos como multiplicativos. El análisis de seguridad y la comparación general entre los esquemas permitió distinguir los beneficios que cada uno ofrece.

Respecto a la seguridad, la firma ECDSA es segura bajo el ataque del mensaje escogido de un falsificador existencial si la función picadillo es resistente a colisiones. El esquema de firma corta ha sido demostrado como infalsificable bajo el supuesto de seguridad del Problema Computacional de Diffie-Hellman (PCDH) y la resistencia a colisiones de la función *map-to-point*.

En la construcción de la firma, tanto ECDSA como la firma corta utilizan el número mínimo de operaciones criptográficas. También, producen firmas de longitud menor que el restos de los esquemas presentados.

A pesar de que la firma ECDSA es la mejor opción en eficiencia para ser uno de los bloques de seguridad de las votaciones electrónicas, el principal inconveniente es el uso de la llave de sesión que, como se verá en el Capítulo 5, abre la posibilidad de crear boletas falsas, en su aplicación a las votaciones electrónicas.

Por lo anterior, el esquema de firma corta fue considerada para ser uno de los bloques criptográficos básicos del esquema de votación electrónica propuesto en esta tesis.

Esquema	# Operación criptográfica	# Ciclos
RSA	1 RSA-privada	51,335,626
	1 RSA-pública	1,822,532
		Total 53,158,158
DSA	3 exponenciaciones	65,351,940
		Total 65,351,940
ECDSA	3 mult. escalares	912,000
		Total 912,000
Firma corta	1 mult. escalar	383,900
	2 H_1	657,900
	2 emparejamientos	4,980,000
		Total 6,021,800
Firma IBE	2 mult. escalares	767,800
	3 emparejamientos	7,470,000
		Total 8,237,800

Tabla 3.4: Comparación de eficiencia entre los esquemas de firma digital.

Capítulo 4

Firma a ciegas

Las firmas digitales son usadas para autenticar al usuario quien firma el mensaje. En algunas aplicaciones es necesario que el signatario desconozca el mensaje, por tanto, una tercera entidad solicita la firma. Este tipo de firmas son denominadas firmas a ciegas y son muy útiles en aplicaciones como votaciones electrónicas y monederos electrónicos.

En este capítulo se analiza el funcionamiento de las primitivas de las firmas a ciegas, sus propiedades y requerimientos.

Además, se propone un esquema novedoso de firma a ciegas definido en la criptografía sobre curvas elípticas, el cual cumple con los requerimientos de seguridad y es más eficiente que el esquema de Jena et al. el cual está definido también sobre curvas elípticas.

Finalmente, se realiza una comparativa de seguridad y de eficiencia de nuestra propuesta junto con nueve esquemas de firma a ciegas propuestos en la literatura. Los resultados presentados en este capítulo fueron obtenidos de [57, 58].

4.1. Definición

Las firmas a ciegas son una clase de firmas digitales producidas para mensajes que se mantienen ocultos para el signatario. Este concepto fue introducido por Chaum en 1982 [20] con la finalidad de garantizar anonimato en los sistemas de pago electrónico. Las firmas a ciegas requieren de dos entidades, el *usuario* quien solicita la firma y el *signatario* quien firma el mensaje.

En [20], Chaum define dos funciones para producir una firma. La primera consiste en una función de firma s conocida sólo por el signatario y su correspondiente función de verificación v, la cual es pública y de la forma $v(s(m)) = \{verdadero/falso\}$, donde m es el mensaje. La segunda es una *función conmutativa* c y su inversa c' tal que $c'(s(c(m))) = s(m)$, ambas conocidas sólo por el usuario que solicita la firma. Una firma a ciegas consta de los siguientes elementos:

Notación

- $\diamond$ $\mathcal{M}$ denota el conjunto de todos los mensajes que pueden ser ocultados.
- $\diamond$ $\mathcal{M}'$ denota el conjunto de todos los mensajes ocultos que pueden ser firmados.
- $\diamond$ $\mathcal{S}'$ denota el conjunto de todas las firmas que pueden producirse a ciegas.
- $\diamond$ $\mathcal{S}$ denota el conjunto de todas las firmas que pueden obtenerse a través de una firma a ciegas.
- $\diamond$ $\mathcal{K}_{\mathcal{S}}$ denota el conjunto de llaves privadas.
- $\diamond$ $\mathcal{K}_{\mathcal{V}}$ denota el conjunto de llaves públicas.
- $\diamond$ $\mathcal{Z}$ denota el conjunto de números enteros denominados factores de opacidad utilizados para ocultar un mensaje.
- $\diamond$ $\mathcal{B} : \mathcal{Z} \times \mathcal{M} \rightarrow \mathcal{M}'$ representa las reglas para ocultar un mensaje.
- $\diamond$ $\mathcal{S}_{\mathcal{E}} : \mathcal{K}_{\mathcal{S}} \times \mathcal{M}' \rightarrow \mathcal{S}'$ representa las reglas para firmar un mensaje oculto por una entidad $\mathcal{E}$.
- $\diamond$ $\mathcal{U} : \mathcal{Z} \times \mathcal{S}' \rightarrow \mathcal{S}$ representa las reglas para obtener una firma a ciegas.
- $\diamond$ $\mathcal{V}_{\mathcal{E}} : \mathcal{K}_{\mathcal{V}} \times \mathcal{S} \times \mathcal{M} \rightarrow \{verdadero, falso\}$ representa las reglas de verificación para una firma producida por $\mathcal{E}$ de forma ciega. Estas reglas son usadas por las entidades que requieran verificar la firma producida por $\mathcal{E}$.

Una firma a ciegas se define como sigue [20]:

Definición 4.1.1. Un esquema de firma a ciegas es el conjunto de algoritmos (*Genera,Oculta, Firma,Descubre,Verifica*), tales que

i. *Genera* es el algoritmo de generación de llaves. Tiene como entrada el parámetro de seguridad ℓ y como salida el par $(k_{\mathcal{S}}, k_{\mathcal{V}}) \in \mathcal{K}_{\mathcal{S}} \times \mathcal{K}_{\mathcal{V}}$, correspondiente a la llave privada y pública, respectivamente.

ii *Oculta* es el algoritmo que transforma el mensaje original en un mensaje oculto a través de la función c. Tiene como entrada $(b, m) \in \mathcal{Z} \times \mathcal{S}'$. b que corresponde a uno o varios factores de opacidad utilizados para ocultar el mensaje m. Tiene como salida el mensaje oculto $m' \in \mathcal{M}'$.

iii. *Firma* es el algoritmo de firma a ciegas. Tiene como entrada $(m', k_{\mathcal{S}}) \in \mathcal{M}' \times \mathcal{K}_{\mathcal{S}}$ y como salida un elemento $s' \in \mathcal{S}'$, el cual es la firma del mensaje m', producida con la llave privada $k_{\mathcal{S}}$.

iv *Descubre* es el algoritmo que transforma la firma del mensaje oculto a la firma del mensaje original a través de la función c'. Tiene como entrada $(b, s') \in \mathcal{Z} \times \mathcal{S}'$ y como salida, la firma a ciegas $s \in \mathcal{S}$ del mensaje m previamente ocultado utilizando el algoritmo Oculta.

iii. *Verifica* es el algoritmo de verificación. Tiene como entrada $(m, s, k_{\mathcal{V}}) \in \mathcal{M} \times \mathcal{S} \times \mathcal{K}_{\mathcal{V}}$ y como salida el valor *verdadero*, lo cual indica una firma válida, o *falso*, en

caso contrario. Se dice que una firma es válida si para cualquier tripleta $(m, s, k_{\mathcal{V}})$ se cumple que:

$$Verifica(m, Firma(m, k_{\mathcal{S}}), k_{\mathcal{V}}) = verdadero$$

para cada $(k_{\mathcal{S}}, k_{\mathcal{V}})$ obtenido del algoritmo *Genera* y para cada $m \in \mathcal{M}$.

4.2. Propiedades y funcionalidad

Una firma a ciegas, además de cumplir con las propiedades de una firma digital, debe satisfacer lo siguiente [50],

Exactitud: la firma puede ser verificada por cualquier entidad y debe ser válida únicamente si la llave pública del signatario es usada en el proceso de verificación de una copia fiel del mensaje original. Es decir, dados m y s genuinos, y la llave pública k_V, la cual es el par de la llave privada que fue usada para firmar m, entonces:

$$Verifica(m, s, k_V) = verdadero$$

Opacidad: conociendo los algoritmos Oculta, Descubre y el mensaje oculto m', el signatario debe ser incapaz de obtener,

$$Descubre(b, m') = m$$

para una o varias b's arbitrarias.

Infalsificable: ninguna entidad debe ser capaz de producir más de una firma válida a partir de la firma del mensaje oculto. Dadas las tuplas almacenadas por el usuario en la producción de n firmas válidas en conjunto con el signatario,

$$(s'_1, m'_1, m'_1, b'_1), ..., (s'_n, m'_n, m'_n, b'_n)$$

Para el usuario, el cual es el solicitante de la firma, dado $y \neq m_i$, es difícil producir,

$$Firma(s'_i, y) = s_{falsa}$$

tal que,

$$Verifica(y, s_{falsa}, k_V) = verdadero$$

No-vinculable: el signatario no tiene la capacidad de distinguir cuál de todos los usuarios le solicitó la firma a ciegas para un determinado mensaje. La información almacenada por el signatario al producir las firmas a ciegas es un conjunto de tuplas $(m'_1, s'_1, ..., m'_n, s'_n)$. Al publicarse los mensajes con sus firmas conoce también $(m_1, s_1, ..., m_n, s_n)$. A pesar de toda la información conocida, para el signatario es imposible vincular si (m'_i, s'_i) corresponde a (m_i, s_i) o si (m_j, s_j) corresponde a (m'_j, s'_j).

La Figura 4.1 muestra el paso de mensajes básico entre el usuario y el signatario para producir una firma. Los algoritmos Oculta y Descubre son utilizados por el usuario, mientras que el algoritmo de firma lo ejecuta el signatario. En la verificación, la cual se muestra en la Figura 4.2, el procedimiento es idéntico a la verificación de una firma digital, donde se requiere el mensaje, la firma y la llave pública del signatario y se produce como salida un mensaje de firma válida o firma rechazada.

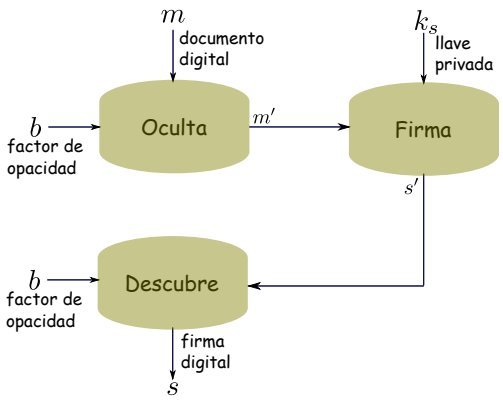

Figura 4.1: Procedimiento de firma a ciegas.

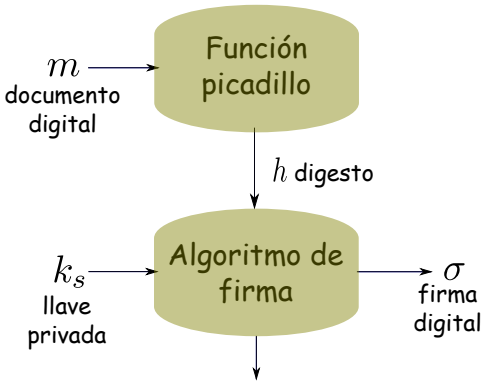

Figura 4.2: Procedimiento de verificación.

4.3. Seguridad

En las firmas a ciegas, la seguridad está definida para dos tipos de adversarios. Primero, el signatario visto como adversario, el cual tendría como objetivo vincular a un usuario con alguna firma producida a ciegas, lo que implica violar las propiedades de no-vinculable y opacidad. Segundo, el usuario visto como adversario cuyo objetivo sería el de producir una firma para un mensaje sin haber establecido comunicación con el signatario exclusivamente en ese mensaje.

Respecto a éste último, las definiciones de seguridad de una firma digital no son aplicables en las firmas a ciegas, ya que la falsificación existencial se produce al permitir al usuario obtener una firma para un mensaje que en teoría jamás fue firmado por el signatario. Recordemos que el mensaje que el signatario firma es el mensaje oculto y en la primitiva "Descubre", el usuario obtiene la firma para el mensaje original.

Pointcheval y Stern [73, 74] propusieron una noción de seguridad donde se permite al usuario obtener n firmas válidas después n de interacciones con el signatario, quedando inhabilitado el usuario para producir una o más firmas después de esas n interacciones.

Como es usual, un adversario tiene distintos modelos de ataques para romper un esquema de firma a ciegas, en [73] se definen los siguientes:

i) Falsificación $(n, n+1)$: para algún entero n, una falsificación $(n, n+1)$ se realiza si un adversario produce $n+1$ firmas después de n interacciones con el signatario.

ii) Falsificación uno-más: para algún entero n y un parámetro de seguridad ℓ, el adversario puede producir $n+1$ firmas válidas después de tener poco más de n interacciones con el signatario. En otras palabras, la "falsificación uno-más" es una falsificación $(n, n+1)$ para un entero n acotado en función de ℓ.

Para alcanzar el éxito en la falsificación, el adversario puede realizar dos tipos de ataques.

i) *Ataque secuencial.* El adversario interactúa secuencialmente con el signatario. El ataque puede ser desarrollado por un usuario, el cual solicita la firma de mensajes ocultos uno después de otro.

ii) *Ataque paralelo.* El adversario interactúa n veces con el signatario. Este ataque es más poderoso, ya que permite al usuario establecer comunicación con el signatario y comenzar una nueva solicitud antes de terminar la primera. Lo anterior sugiere que el ataque puede ser implementado por varios usuarios habilitados para solicitar firmas al signatario al mismo tiempo.

Los argumentos de seguridad para las firmas a ciegas se basan en problemas computacionalmente difíciles de resolver. Por tanto, el adversario es un algoritmo eficiente que se supone puede resolver un problema difícil, utilizando normalmente la reducción de problemas.

En las reducciones es importante simular adecuadamente las interacciones del adversario con las demás entidades. Recordemos que en las firmas digitales, el adversario tiene acceso al algoritmo de firma a través de un oráculo aleatorio. En el caso de las firmas a ciegas, la producción de la firma requiere de la interacción de dos entidades, por tanto, el oráculo aleatorio puede simular tanto al usuario en caso de que el adversario sea el signatario, o simular al signatario en caso de que el adversario sea el usuario. Para mayor información, el lector interesado puede consultar [73, 74].

4.4. Esquemas de firmas a ciegas propuestos

La gran mayoría de los esquemas de firma a ciegas que han sido propuestos han sido modificaciones de esquemas de firma digital. La primera firma práctica propuesta en este contexto fue la firma presentada por Chaum [18], donde usa una ingeniosa modificación de la firma digital RSA. Poco después, Chaum presentó una nueva propuesta basada en la firma digital DSA [19]. Desde entonces, se han publicado varios esquemas de firma a ciegas. En esta sección listaremos los esquemas que hemos considerado los mejores publicados en la literatura.

En [15], Camenisch et al. propusieron dos esquemas de firma a ciegas. Uno de ellos basado en la firma digital DSA y el otro es una modificación del esquema de firma

digital de Ryberg y Rueppel [70]. Cao y Liu en [16] usaron una modificación de la firma digital de Zhu [96] y basaron la seguridad de su esquema en el problema computacional RSA-fuerte.

Jena et al. [43] presentaron un esquema de firma a ciegas basado en la criptografía de curvas elípticas. Boldyreva [11] presentó un esquema muy elegante usando emparejamientos bilineales. Por último, Gao et al. [35] propusieron un esquema basado en la identidad usando emparejamientos.

En las siguientes secciones son analizados y clasificados los esquemas de firma a ciegas mencionados de acuerdo al grupo donde están definidos, comúnmente grupos multiplicativos sobre enteros (RSA,DSA), grupos aditivos sobre curvas elípticas (CCE) o grupos aditivos transformados a grupos multiplicativos definidos sobre campos primos (emparejamientos bilineales). Se presenta también un análisis de seguridad y de eficiencia de los esquemas de firmas a ciegas. Para mayor comodidad del lector, los algoritmos de la mayoría de los esquemas mencionados se encuentran en el Apéndice A, presentando únicamente en este capítulo los algoritmos de los esquemas de firma a ciegas que se utilizarán en los capítulos posteriores de esta tesis.

4.4.1. Firma a ciegas de Chaum

El esquema propuesto por Chaum [18] fue el primer esquema de firma a ciegas reportado en la literatura y está basado en la firma digital RSA. Los algoritmos 4.1, 4.2, 4.3 y 4.4 muestran los pasos para producir una firma a ciegas, mientras que el algoritmo 4.5 muestra la verificación, la cual es idéntica a la verificación de la firma digital RSA.

Como se puede notar en 4.2 y 4.4, el esquema utiliza un factor de opacidad b para ocultar el mensaje y para descubrir la firma. Por otra parte, sólo dos mensajes son intercambiados entre el usuario y el signatario para producir la firma que consta del valor entero s.

Respecto a la seguridad, como en RSA, la firma a ciegas de Chaum requiere de una función picadillo de la forma $H : \{0,1\}^* \rightarrow \{1,2,...,N-1\}$ para evitar la falsificación $(n, n+1)$ por parte del usuario. Por otra parte, el factor de opacidad evita que el signatario obtenga el mensaje original a partir del mensaje oculto. Cuando la firma es publicada, dado que el usuario al descubrir la firma retira el factor de opacidad, el signatario no tiene forma de relacionar el mensaje oculto $h' \equiv hb^e \bmod N$ con el mensaje original m.

Algoritmo 4.1: Genera (signatario), Chaum [18]

Entrada: Parámetro de seguridad ℓ.
Salida: Llaves (d, e, N).

1 Elegir aleatoriamente dos números primos p y q, de $\ell/2$ bits de longitud;
2 $N = pq$;
3 $\phi(N) = (p-1)(q-1)$, con $\text{mcd}(e, \phi(N)) = 1$;
4 Elegir aleatoriamente un número pequeño e;
5 Calcular $d \equiv e^{-1} \bmod \phi(N)$;
6 **return** (d, e, N)

Algoritmo 4.2: Oculta (usuario), Chaum [18]

Entrada: El mensaje m.
Salida: El mensaje oculto h', el factor de opacidad b.

1 Calcular $h = H(m)$;
2 Seleccionar aleatoriamente $b \in \mathbb{Z}_N^*$;
3 $h' \equiv hb^e \bmod N$;
4 **return** (h', b)

Algoritmo 4.3: Firma (signatario), Chaum [18]

Entrada: El mensaje oculto h', la llave privada del signatario (d, N).
Salida: La firma a ciegas s'.

1 Calcular $s' \equiv (h')^d \equiv (hb^e)^d \bmod N$;
2 **return** s'

Algoritmo 4.4: Descubre (usuario), Chaum [18]

Entrada: La firma a ciegas s', el factor de opacidad b.
Salida: La firma s.

1 $s \equiv s'b^{-1} \equiv (h^d)bb^{-1} \bmod N$;
2 **return** s

Algoritmo 4.5: Verifica, Chaum [18]

Entrada: La firma s, el mensaje m, la llave pública (e, N).
Salida: {Válida/Rechazada}.

```
1 h = H(m);
2 h' ≡ s^e mod N;
3 if h = h' then
4     return Válida;
5 else
6     return Rechazada;
7 end
```

4.4.2. Firma a ciegas de Cao y Liu

Cao y Liu [16] propusieron un esquema basado en el supuesto de seguridad RSA-fuerte, el cual garantiza que dado un módulo N y un número aleatorio $c \in \mathbb{Z}_N^*$ es computacionalmente intratable encontrar $a, b \in \mathbb{Z}_N$, $b \geq 2$ tales que $a^b \equiv c \bmod N$.

Los Algoritmos A.1, A.2, A.3, A.4 y A.5 muestran la generación de llaves, producción y verificación de la firma a ciegas de Cao y Liu.

En la generación de llaves, el signatario selecciona dos números primos p y q de igual longitud y ambos forman la llave privada, mientras que la tripleta (N, X, g) corresponde a la llave pública del signatario, donde $N = pq$ es el módulo RSA y el par (X, g) son generadores de residuos cuadráticos en $\mathbb{Z}_N$.

Para ocultar el mensaje, el usuario utiliza dos funciones picadillo $H_1 : \{0,1\}^* \rightarrow \{0,1\}^{768}$, y $H : \{0,1\} \rightarrow \{1, 2, ..., N-1\}$, dos factores de opacidad y los generadores de residuos cuadráticos.

La producción de la firma requiere del intercambio de dos mensajes entre las entidades. Una de las principales diferencias con respecto al esquema de Chaum es que la firma consiste de dos valores denotados como e y y.

De acuerdo a la prueba de seguridad presentada por los autores, el esquema es seguro en contra del ataque de mensaje escogido y se satisfacen las primitivas de opacidad y no-vinculable.

4.4.3. Esquemas de firma a ciegas de Camenisch

Camenisch et al. propusieron dos esquemas de firma a ciegas en [15]. El primero de ellos es una variación de la firma digital DSA; el segundo está basado en la firma digital propuesta por Nyberg y Rueppel [70].

En la generación de llaves es la misma para ambas propuestas. Los Algoritmos A.6, A.7, A.8, A.9 y A.10 muestran la generación, producción y verificación de la primera propuesta de firma a ciegas de Camenisch et al.

En la generación de llaves, el signatario debe generar una llave de sesión por cada solicitud de firma a ciegas, lo que implica un mensaje adicional entre las entidades. Por otra parte, el usuario usa dos factores de opacidad para ocultar el mensaje, los cuales no son retirados al descubrir la firma, debido al hecho de que la llave de sesión k es usada para producir los valores de la firma r y s, y los factores de opacidad son utilizados para ocultar y proteger la llave de sesión. De tal manera, que el signatario no pueda relacionar la firma (r, s) con alguna k almacenada previamente por él. En la firma del mensaje, el proceso es muy similar al algoritmo de firma digital DSA.

La segunda propuesta de Camenisch et al. está basada en la firma digital de Nyberg y Rueppel y tiene la particularidad de ser una firma con recuperación de mensaje. Tal propiedad permite obtener el mensaje a través de la firma digital, por tanto, no es necesario que el mensaje sea enviado junto con la firma para realizar la verificación. Los Algoritmos A.11, A.12, A.13 y A.14 muestran los pasos para firmar y verificar la firma a ciegas con recuperación de mensaje.

Desafortunadamente, los autores no presentaron pruebas de seguridad para sus propuestas. Hecho que probablemente motivó a varios autores para encontrar alguna debilidad en ambos esquemas, aunque hasta ahora no han tenido éxito.

En particular, varios autores intentaron atacar la primera propuesta con el objetivo de encontrar alguna relación entre un usuario y alguna firma. En 1995, Harn [38] afirmó que era posible relacionar el par (usuario, firma) y que por tanto, no cumplía con la propiedad de no-vinculable. Sin embargo, en el mismo año, Horster et al. [41] probaron que la afirmación de Harn era incorrecta. Algo similar sucedió con Lee et al. quienes en el 2002, aseguraron en [52] que la firma era rastreable y propusieron dos esquemas de firma digital basadas en el PLD, argumentando que sus propuestas corregían la debilidad encontrada. No obstante, Wu y Wang [94] encontraron errores en las propuestas de Lee.

Finalmente, Wu [93] demostró en el 2006, que la primera propuesta de Camenisch et al. basada en una variación de la firma digital DSA, es infalsificable y no-vinculable.

4.4.4. Firma a ciegas de Jena, Kumar y Majhi sobre curvas elípticas

Jena et al. propusieron en [43] un esquema basado en el problema del logaritmo discreto sobre curvas elípticas. El esquema es similar a la primera propuesta de Camenisch et al. pero en su versión sobre curvas elípticas.

Como la firma digital ECDSA, la firma a ciegas de Jena et al. utiliza tanto operaciones modulares en enteros como operaciones con puntos sobre la curva elíptica, pero produce una firma que consta de dos números enteros. Los Algoritmos A.15, A.16, A.17, A.18 y A.19 muestran la generación, producción y verificación de la firma a ciegas de Jena et al.

En la generación de llaves, la llave pública del signatario es un punto que satisface una curva elíptica. Como en la versión de Camenisch et al., la propuesta de Jena et al.

requiere el uso de una llave de sesión generada de forma única por cada solicitud de firma a ciegas hecha al signatario.

Para ocultar el mensaje, se utilizan dos factores de opacidad y la llave de sesión generada por el signatario. Las operaciones modulares son principalmente multiplicaciones en $\mathbb{Z}_q^*$ y una inversión en el mismo campo primo.

La firma es producida calculando una operación modular. En el descubrimiento de la firma los factores de opacidad permanecen en la firma para evitar que la llave de sesión sea vinculada con el solicitante de la firma.

Los autores definen la firma con el par (r, s). Sin embargo, con el objetivo de verificar la firma, la entidad verificadora debe tomar un punto R para calcular una multiplicación escalar, por tanto, r es tomado como la coordenada x del punto R.

Respecto a la seguridad, los autores afirman que la firma a ciegas cumple con todos los requerimientos, sin embargo la demostración de seguridad no fue considerada en la publicación de la propuesta.

4.4.5. Firma a ciegas de Boldyreva

Boldyreva [11] propuso en el 2003, una firma a ciegas basada en emparejamientos bilineales. Los parámetros de dominio son los siguientes. Sean r un número primo, dos grupos aditivos $\mathbb{G}_1$ y $\mathbb{G}_2$, un grupo multiplicativo $\mathbb{G}_T$ todos los grupos de orden r, la función *map-to-point* $H_1 : \{0,1\} \rightarrow \mathbb{G}_1$ y la función de emparejamiento bilineal $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_2 \rightarrow \mathbb{G}_T$.

La generación de llaves la cual se muestra en el Algoritmo 4.6 es similar a los esquemas basados en la criptografía sobre curvas elípticas.

El ocultamiento del mensaje requiere de la función H_1 y de un factor de opacidad b, como se muestra en el Algoritmo 4.7. El Algoritmo 4.8 muestra que el signatario calcula únicamente una multiplicación escalar para firmar a ciegas. Como en el caso de la firma a ciegas de Chaum, la propuesta de Boldyreva retira completamente el factor de opacidad como se indica en el Algoritmo 4.9.

Finalmente, el esquema usa la propiedad de bilinealidad de los emparejamientos para resolver el problema de decisión de Diffie-Hellman (véase tabla 3.1) para garantizar la validez de la firma en el algoritmo de verificación 4.10.

La multiplicación escalar predomina como la operación principal en las primitivas para ocultar y firmar el mensaje y descubrir la firma, dejando a los emparejamientos bilineales sólo en la primitiva de verificación.

La firma producida es el punto S, el cual es generado con tan solo dos mensajes entre las entidades. Para garantizar la seguridad de la firma, se requiere la función especial denominada *map-to-point*, la cual proyecta el mensaje hacia un punto en la curva elíptica.

Algoritmo 4.6: Genera (signatario), Boldyreva [11]

Entrada: Curva elíptica E, punto base P, orden r.
Salida: Las llaves (d, Q).

1 Seleccionar aleatoriamente $d \in \mathbb{Z}_r^*$;
2 $Q = dP$;
3 **return** (d, Q)

Algoritmo 4.7: Oculta (usuario), Boldyreva [11]

Entrada: El mensaje m.
Salida: El mensaje oculto $\hat{M}$, el factor de opacidad b.

1 $M = H_1(m)$;
2 Seleccionar aleatoriamente $b \in \mathbb{Z}_r^*$;
3 $\hat{M} = bM$;
4 **return** $(\hat{M}, b)$

Algoritmo 4.8: Firma (signatario), Boldyreva [11]

Entrada: El mensaje oculto $\hat{M}$, la llave privada d.
Salida: La firma a ciegas $\hat{S}$.

1 $\hat{S} = d\hat{M}$;
2 **return** $\hat{S}$

Algoritmo 4.9: Descubre (usuario), Boldyreva [11]

Entrada: La firma a ciegas $\hat{S}$, el factor de opacidad b.
Salida: La firma S.

1 $S = b^{-1}\hat{S}$;
2 **return** S

Algoritmo 4.10: Verifica, Boldyreva [11]

Entrada: La firma S, el mensaje m, la llave pública Q.
Salida: {Válida/Rechazada}.

1 **if** $\hat{e}(P, S) = \hat{e}(Q, H_1(m))$ **then**
2 **return** *Válida*;
3 **else**
4 **return** *Rechazada*;
5 **end**

Boldyreva demuestra que su esquema es seguro a través de una reducción al problema denominado *chosen-target CDH*, definido como sigue.

Dados un número primo r, un grupo abeliano $\mathbb{G} = \langle P \rangle$ de orden r y un elemento aleatorio $d \in \mathbb{Z}_r^*$ tales que $Q = dP$, existe un oráculo $\mathcal{O}^H$ que tiene como salida puntos aleatorios en $\mathbb{G}$ y otro oráculo $\mathcal{O}^d$ que realiza la inversión del PLD, es decir, a partir de Q y P obtiene d en la ecuación $Q = dP$, con la restricción de que el número de consultas a $\mathcal{O}^d$ es estrictamente menor al número de consultas a $\mathcal{O}^H$.

Al adversario $\mathcal{B}$ se le permite realizar consultas a los oráculos $\mathcal{O}^H$ y $\mathcal{O}^d$. Dado el número de consultas q_H y q_d hechas a los oráculos $\mathcal{O}^H$ y $\mathcal{O}^d$, respectivamente. La ventaja del adversario atacando el problema *chosen-target CDH* está definida como la probabilidad de $\mathcal{B}$ para producir ℓ pares $((v_1, j_1), ..., (v_\ell, j_\ell))$, tales que $v_i = dZ_{j_i}$, donde v_i son respuestas dadas por el oráculo $\mathbb{O}^d$, Z_i son respuestas dadas por el oráculo $\mathbb{O}^H$ y además, para todo $1 \leq i \leq \ell$ existe $1 \leq j_i \leq q_t$, con todos los v_i distintos y $q_d < q_H$, y se denota como $Adv_{\mathbb{G}}^{ct-cdh}(B)$.

Como puede observarse, si el adversario hace únicamente una consulta al oráculo $\mathcal{O}^H$ entonces el problema *chosen-target CDH* es equivalente al PCDH.

La demostración se inicia con la suposición de que el problema *chosen-target CDH* es computacionalmente intratable en todos los grupos donde el problema computacional de Diffie-Hellman (PCDH) es intratable.

La definición de seguridad para el esquema de firma a ciegas Boldyreva (BS) es como sigue.

Definición 4.4.1. El esquema BS es seguro en contra de un falsificador uno-más $\mathcal{A}$ utilizando el ataque de mensaje escogido bajo el modelo del oráculo aleatorio, si existe un adversario $\mathcal{B}$ para el problema *chosen-target CDH* tal que

$$Adv_{BS[\mathbb{G}]}^{falsifica}(A) = Adv_{\mathbb{G}}^{ct-cdh}(B)$$

La demostración se desarrolla como un juego entre el falsificador $\mathcal{A}$ y el adversario $\mathcal{B}$ que manipula los oráculos aleatorios. Ambas entidades son algoritmos de tiempo polinomial.

El juego comienza con las consultas hechas por $\mathcal{A}$ primero al oráculo $\mathcal{O}^H$ mismas que son respondidas a través de $\mathcal{B}$. Después de obtener varios puntos aleatorios, $\mathcal{A}$ consulta al oráculo de firma. Para lo cual $\mathcal{B}$ utiliza a $\mathbb{O}^d$. Al final de las consultas, $\mathcal{A}$ consigue una lista de ℓ pares $((M_1, \sigma_1)...(M_\ell, \sigma_\ell))$. Mientras que $\mathcal{B}$, para cada $1 \leq i \leq \ell$, encuentra un M_i en una lista generada de las respuestas del oráculo $\mathbb{O}^H$. Sea j_i el índice del par encontrado, $\mathcal{B}$ tiene como salida la lista $(\sigma_1, j_1)...(\sigma_\ell, j_\ell)$. Lo que significa que por cada consulta hecha por $\mathcal{A}$ a los oráculos aleatorios, $\mathcal{B}$ busca un par que satisfaga la ecuación $\sigma_i = dM_{j_i}$, donde σ_i es la firma del mensaje M_{j_i} y se satisface que la firma es válida. Evidentemente, $\mathcal{B}$ tendrá éxito si $\mathcal{A}$ tiene éxito, de tal manera que la falsificación se produce si

$$Adv_{BS[\mathbb{G}]}^{falsifica}(A) = Adv_{\mathbb{G}}^{ct-cdh}(B)$$

.

4.4.6. Firma a ciegas de Gao et al.

Gao et al. [35] desarrollaron un esquema de firma a ciegas basado en la identidad. Los parámetros de dominio consisten en dos grupos aditivos $\mathbb{G}_1$ y $\mathbb{G}_2$ y un grupo multiplicativo $\mathbb{G}_T$, todos de orden r, la función $H_1 : \{0,1\} \rightarrow \mathbb{G}_1$ y la función de emparejamiento $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_2 \rightarrow \mathbb{G}_T$.

El esquema basa su seguridad en el problema computacional de inversión de Diffie-Hellman (BDHI) definido como el problema de calcular un punto $S \in \mathbb{G}_1$ tal que $\hat{e}(P, S) = h$ con $h \in \mathbb{G}_T$.

A diferencia de la firma de Zhang y Kim, Gao et al. no requieren de una llave de sesión. Los Algoritmos A.20, A.21, A.22 y A.23 muestran los pasos para producir y verificar una firma.

La firma consiste en tres puntos y la principal desventaja de esta propuesta es que se requiere calcular cuatro emparejamientos bilineales para ocultar el mensaje y cuatro funciones más para la verificación de la firma. Por tanto, este esquema es el más ineficiente de los esquemas listados.

Por último, Gao et al. demostraron que el esquema cumple con el requerimiento de opacidad y que es seguro en contra de la falsificación uno-más.

4.5. Propuesta de firma ciegas definida sobre curvas elípticas

La propuesta de Camenisch et al. basada en la firma de Ryberg y Rueppel puede ser eficientemente traducida a una versión sobre puntos que pertenecen a una curva elíptica definida sobre un campo primo.

Nuestra propuesta sobre curvas elípticas tiene el mismo número de factores de opacidad y el mismo número de mensajes entre las entidades que el segundo esquema de Camenisch et al. utilizando enteros.

En la generación de llaves, ilustrada en el Algoritmo 4.11, se utiliza una curva elíptica E definida sobre un campo binario $\mathbb{F}_{2^m}$ y un punto generador $P \in E(\mathbb{F}_{2^m})$ de orden n. La llave privada es el número aleatorio $d \in \mathbb{Z}_n^*$ y la llave pública el punto $Q = dP$.

Adicionalmente, para cada solicitud de firma a ciegas, el signatario genera una llave de sesión $k \in \mathbb{Z}_n^*$ y el punto $\hat{R}$ mismos que serán utilizados por el usuario para ocultar el mensaje.

Algoritmo 4.11: Genera (signatario), López et al. [57]

Entrada: Curva elíptica E, punto base P, orden n.
Salida: Las llaves (d, Q).

1 Seleccionar aleatoriamente $d \in \mathbb{Z}_n^*$;
2 $Q = dP$;
3 **return** (d, Q)

Para ocultar el mensaje m, como se muestra en el Algoritmo 4.12, el usuario usa el punto $\hat{R}$ y dos factores de opacidad $a, b \in \mathbb{Z}_n^*$. A pesar de que la versión original de Ryberg y Rueppel tiene la propiedad de la recuperación del mensaje y por tal motivo no se utiliza la función picadillo H, en nuestra propuesta el punto R es calculado utilizando $\hat{R}$, los factores de opacidad y el picadillo del mensaje m. Finalmente, $\hat{h}$ corresponde a la coordenada x del punto R ocultada con los factores de opacidad con una operación módulo n.

Algoritmo 4.12: Oculta (usuario), López et al. [57]

Entrada: El mensaje m.
Salida: El mensaje oculto $\hat{h}$, los factores de opacidad (a, b) y el punto R.

1 El signatario: para cada mensaje, seleccionar aleatoriamente $k \in \mathbb{Z}_n^*$;
2 Calcular $\hat{R} = kP$ {El signatario envía al usuario $\hat{R}$};
3 El usuario: para cada mensaje, seleccionar aleatoriamente $a, b \in \mathbb{Z}_n^*$;
4 $h = H(m)$;
5 $(x_R, y_R) = R = (h + a)P + b\hat{R}$;
6 $r \equiv x_R \bmod n$;
7 $\hat{h} \equiv rb^{-1} \bmod n$;
8 **return** $(\hat{h}, R, a, b)$

El signatario al recibir el mensaje oculto $\hat{h}$, lo firma (Algoritmo 4.13) utilizando la llave de sesión k y su llave privada d. Es importante resaltar que la producción de la firma es idéntica a la versión con enteros de Camenisch et al. mostrada en el Algoritmo A.12.

Algoritmo 4.13: Firma (signatario), López et al. [57]

Entrada: El mensaje oculto $\hat{h}$, la llave privada d, la llave de sesión k.
Salida: La firma a ciegas $\hat{s}$.

1 $\hat{s} \equiv \hat{h}d + k \bmod n$;
2 **return** $\hat{s}$

En el descubrimiento de la firma, en el Algoritmo 4.14, el usuario no retira los factores de opacidad a y b para evitar que el signatario pueda vincularlo con la firma a ciegas a través de la llave de sesión. La firma producida entre el signatario y el usuario es el par (R, s) para el mensaje m.

Algoritmo 4.14: Descubre (usuario), López et al. [57]

Entrada: La firma oculta $\hat{s}$, los factores de opacidad (a, b).
Salida: El valor s.

1 $s \equiv \hat{s}b + a \bmod n$;
2 **return** s

En el Algoritmo 4.15, se muestra la verificación de la firma (R, s). A diferencia de la segunda propuesta de firma a ciegas de Camenisch, nuestro esquema sí requiere el mensaje para la verificación de la firma, eliminando así la característica de ser un esquema de firma con recuperación de mensaje.

Algoritmo 4.15: Verifica, López et al. [57]

Entrada: La firma $(R = (x_R, y_R), s)$, el mensaje m, la llave pública Q.
Salida: {Válida/Rechazada}.

1 $r \equiv x_R \bmod n$;
2 $h = H(m)$;
3 $T = -sP + rQ + R$;
4 **if** $hP = T$ **then**
5 **return** *Válida*;
6 **else**
7 **return** *Rechazada*;
8 **end**

4.5.1. Análisis de requerimientos

Exactitud
La firma puede ser verificada por cualquier entidad y es válida únicamente si la ecuación siguiente se satisface.

$$\begin{aligned} -sP + rQ + R &= -(\hat{s}b + a)P + rdP + (h + a)P + b\hat{R} \\ &= -\hat{h}dbP - kbP - aP + rdP + hP + aP + bkP \\ &= -rb^{-1}dbP + rdP + hP \\ &= -rdP + rdP + hP \\ &= hP \end{aligned}$$

Opacidad
En la primitiva de firma, el signatario no puede obtener el valor del mensaje original a partir de $\hat{h}$, debido a que no se encuentra explícitamente contenido en la ecuación $\hat{h} \equiv rb^{-1} \bmod n$, indicada en la línea 5 del Algoritmo 4.12. Aun cuando el signatario logre por algún medio obtener o adivinar el par (a, b) correcto, el mensaje está protegido por el PLD en curvas elípticas por la ecuación $R = (h + a)P + b\hat{R} = hP + aP + b\hat{R}$ que conocidos los puntos aP y $b\hat{R}$, queda hP que es de la forma $M = hP$ donde es únicamente conocido P mientras que h y M toman valores aleatorios según la manipulación del signatario.

No-vinculable
Dada la firma (R_j, s_j, m_j) construida con las siguientes ecuaciones:

$$\begin{aligned}
&\hat{R}_j = k_j P \\
&h_j = H(m_j) \\
&R_j = h_j P + a_j P + b_j \hat{R}_j \\
&r_j = x_{R_j} \bmod n \\
&\hat{h}_j = r_j b_j^{-1} \bmod n \\
&\hat{s}_j = \hat{h}_j d + k_j \bmod n \\
&s_j = \hat{s}_j b_j + a_j \bmod n \\
&\text{verificación} \\
&h_j P = -s_j P + r_j Q + R_j
\end{aligned}$$

Cuando (m_j, R_j, s_j) es publicada, el signatario busca en las tuplas almacenadas $(k_1, \hat{R}_1, \hat{h}_1, \hat{s}_1)$, ..., $(k_\ell, \hat{R}_\ell, \hat{h}_\ell, \hat{s}_\ell)$ de las ℓ firmas a ciegas producidas. Para cualquier tupla $(k_i, \hat{R}_i, \hat{h}_i, \hat{s}_i)$, despejando b y a de las ecuaciones $\hat{h} = rb^{-1} \bmod n$ y $s = \hat{s}b + a \bmod n$, el signatario calcula:

$$b = \hat{h}_i^{-1} r_j \bmod n$$

$$a = s_j - \hat{s}_i b \bmod n$$

y verifica la siguiente ecuación:

$$\begin{aligned}
-s_j P + r_j Q + R_j &= -(\hat{s}_j b + a)P + r_j Q + h_j P + aP + b\hat{R}_j \\
&= -\hat{s}_j bP - aP + r_j Q + h_j P + aP + b\hat{R}_j \\
&= -\hat{s}_j bP + r_j Q + h_j P + b\hat{R}_j \\
&= -\hat{h}_j dbP - k_j bP + r_j Q + h_j P + b\hat{R}_j \\
&= -\hat{h}_j bQ - b\hat{R}_j + r_j Q + h_j P + b\hat{R}_j \\
&= -\hat{h}_j \hat{h}_i^{-1} r_j Q + r_j Q + h_j P
\end{aligned}$$

si $\hat{h}_i = \hat{h}_j$ entonces la igualdad se cumple y se considera que el mensaje m_j corresponde al registro k_i. Lo que significa que para cualquier i y j existe un valor de $b \in \mathbb{Z}_n^*$ que satisface la ecuación $\hat{h}_i = r_j b^{-1}$. De tal manera que es posible relacionar el mismo valor de b para distintos pares (i, j), evitando así distinguir cual de las tuplas corresponde a la firma (m_j, R_j, s_j).

Infalsificable
En la verificación de la firma, el objetivo para un falsificador existencial es lograr satisfacer $H(m)P = -s_f P + r_f Q + R_f$ para la firma falsa (R_f, s_f) del mensaje m.

$$\begin{aligned} -sP + rQ + R &= -sP + rdP + H(m)P + aP + bk_2P \\ &= (-s + rd + H(m) + a + bk_2)P \\ &= H(m)P + (-s + rd + a + bk_2)P \end{aligned}$$

si $-s+rd+a+bk_2 = 0$ entonces la verificación es válida. Dado que los valores importantes son s y r que corresponderán a la firma falsa entonces la ecuación se reduce a $-s+rd = 0$, para conseguir la siguiente igualdad:

$$\begin{aligned} -s + rd &= 0 \\ rd &= s \\ d &= r^{-1}s \\ dP &= r^{-1}sP \\ Q &= r^{-1}sP \end{aligned}$$

lo que implica resolver el PLDCE para la llave pública $Q = dP$.

4.6. Comparación general

La seguridad de una firma a ciegas está definida por el grado de opacidad, de confianza y de seguridad.

El grado de opacidad se refiere, primero, a que el mensaje debe estar oculto del signatario al momento de ser firmado y segundo, el anonimato del usuario debe garantizarse una vez que la firma sea publicada, es decir, que la firma a ciegas no pueda ser vinculada con el usuario que la solicitó. Los factores de opacidad juegan un rol muy importante en las firmas a ciegas, ya que a través de ellos se oculta el mensaje y se evita la relación entre la firma y el solicitante. En la Tabla 4.1 se puede apreciar que los esquemas de Chaum, Boldyreva y Gao et al. requieren de un factor de opacidad y tienen la característica de que lo usan únicamente para ocultar el mensaje, mientras que el resto de los esquemas necesitan dos factores los cuales deben mantenerse en la firma resultante para evitar que el signatario relacione la llave de sesión correspondiente a esa

firma con algún usuario. Cabe aclarar que todos los esquemas que usan dos factores de opacidad generan una llave de sesión por firma producida.

Esquema	Factores de opacidad	Supuesto de seguridad	Demostración de seguridad
Chaum [1982]	1	PFI	Sí
Cao y Liu [2007]	2	RSA-fuerte	Sí
Camenisch et al. (DSA) [1994]	2	PLD	No
Camenisch et al. (Nyberg y Rueppel) [1994]	2	PLD	No
Boldyreva [2003]	1	PCDH	Sí
Gao et al. [2007]	1	PIDH	Sí
Jena et al. [2008]	2	PLDCE	No
Nuestra propuesta [2008]	2	PLDCE	Sí

Tabla 4.1: Comparación general de los esquemas de firmas a ciegas.

El nivel de confianza es respecto a la producción de la firma para garantizar que fue producida a partir de la interacción entre un usuario y el signatario, por tanto, debe ser resistente a los ataques de falsificación. En la Tabla 4.1 se muestra también el supuesto de seguridad en el que cada firma basa su seguridad y si fue o no desarrollada la demostración de seguridad que garantiza que el esquema es resistente a los ataques de falsificación uno-más y se puede apreciar que los esquemas propuestos por Camenisch et al. y sus versiones sobre curvas elípticas son las que no presentan demostración de seguridad.

Por último, el nivel de seguridad ℓ es el número aproximado de pasos 2^{ℓ} que requiere un algoritmo para resolver un problema computacionalmente intratable. Actualmente, el valor recomendado de ℓ puede ser de 80, 112, 128 y 196 bits. Los parámetros de dominio para todos los esquemas fueron definidos en esta tesis con un nivel de seguridad de 128 bits y se muestran en la Tabla 4.2. Para mayor claridad, los esquemas se presentan divididos de acuerdo al grupo donde se encuentra definida su aritmética.

La Tabla 4.3 muestra el número de mensajes que requiere cada esquema para producir una firma. Siguiendo la línea de un nivel de seguridad de 128 bits, la tabla también ilustra las características de cada firma producida, como el número de elementos por firma, la longitud en bits de los valores de la firma y cuál de la primitivas es la más costosa. Como puede apreciarse, la firma de Boldyreva definida para emparejamientos asimétricos produce una firma de tan sólo 254 bits con tan sólo dos interacciones entre

las entidades, seguida de la firma de Jena et al. basada en curvas elípticas.

Algo notorio en la última columna de la tabla es que los esquemas basados en emparejamientos son más costosos en la verificación de la firma. La razón de este hecho se debe a el cálculo de los emparejamientos, sin embargo, se han propuestos algoritmos altamente eficientes para calcular una función de emparejamiento, con lo cual, es posible, disminuir el costo computacional de estos esquemas.

Esquemas	Parámetros	Longitud en bits
Chaum, Cao y Liu	$\mathbb{Z}_N$	$\|\mathrm{N}\| = 3072$ bits
Camenisch et al.	$\mathbb{Z}_p$ y $\mathbb{Z}_q$	$\|p\| = 3072, \|q\| = 256$
Jena et al. y Nuestra propuesta	$\|\langle P\rangle\| = n$, $P \in E(\mathbb{F}_{2^m})$	$n = 251$ bits
Boldyreva y Gao et al.	$\|\langle P\rangle\| = r$, $P \in E(\mathbb{F}_p)$	$r = 254$ bits
	$\mathbb{F}_{p^k}$	$k = 12$

Tabla 4.2: Seguridad de 128-bits para cada firma digital.

	# Mensajes	Firma producida	Longitud en bits	Primitiva más costosa
Chaum	2	(s)	(3072)	Firma
Cao y Liu	2	(e, y)	(513,3072)	Firma
Camenisch et al. propuesta 1	3	(r, s)	(256,256)	Firma
Camenisch et al. propuesta 2	3	(r, s)	(3072,256)	Oculta
Boldyreva	2	(S)	(254,1)	Verifica
Gao et al.	2	(A, B, C)	((254,1),(254,1),(254,1))	Descubre
Jena et al.	3	(r, s)	(251,251)	Oculta
Nuestra propuesta	3	(R, s)	((251,1),251)	Oculta

Tabla 4.3: Comparación general de los esquemas de firmas a ciegas.

4.7. Comparación de eficiencia

Con la finalidad de comparar el desempeño de los esquemas presentados, se clasificaron de acuerdo al grupo abeliano donde se encuentra definida su aritmética, como se indica en la Tabla 4.4. La implementación fue desarrollada en un procesador Intel Core 2 Quad @2.4Ghz y los algoritmos utilizados para cada operación criptográfica se muestran en el Capítulo 2.

Enteros	Puntos sobre una curva elíptica	Emparejamientos
Chaum Cao y Liu Camenisch et al.	Jena et al. Nuestra propuesta	Boldyreva Gao et al.

Tabla 4.4: Algoritmos de firma a ciegas.

4.7.1. Firmas a ciegas definidas sobre grupos multiplicativos

La Tabla 4.5 compara el desempeño de dos esquemas de firma a ciegas definidos sobre números enteros. Las propuestas de Chaum y de Cao y Lui usan números enteros en el anillo $\mathbb{Z}_N$, donde N es el producto de los números primos p y q. Por otra parte, la Tabla 4.6 muestra los dos esquemas propuestos por Camenisch et al. Todos los esquemas muestran que la primitiva de ocultamiento tiene un costo desdeñable. Sin embargo el esquema de Chaum tiene la ventaja de que únicamente calcula una operación criptográfica por primitiva. De tal manera que la firma a ciegas de Chaum es más flexible que la propuesta por Cao y Liu.

En las firmas a ciegas de Camenisch et al. se requiere de una exponenciación en la primitiva de firma. No obstante, la operación es calculada antes de ocultar el mensaje a través de la llave de sesión producida por el signatario.

	Chaum		Cao-Liu	
	Operación	# Ciclos	Operación	# Ciclos
Oculta	1 RSA-pública	1,822,532	2 RSA-pública	3,645,064
Firma	1 RSA-privada	51,335,626	1 RSA-privada	51,335,626
Descubre	–	–	1 RSA-pública	1,822,532
Verifica	1 RSA-pública	1,822,532	2 RSA-pública	3,645,064
Total	3	54,980,690	6	60,448,286

Tabla 4.5: Esquemas RSA con módulo $N = 3072$ bits.

	Camenisch et al. 1		Camenisch et al. 2	
	Operación	# Ciclos	Operación	# Ciclos
Oculta	2	43,567,960	2	43,567,960
Firma	1	21,783,980	1	21,783,980
Descubre	–	–	–	–
Verifica	3	65,351,940	2	43,567,960
Total	6	130,703,880	5	108,919,900

Tabla 4.6: Esquemas de Camenisch et al. con módulos $p = 3072$ y $q = 256$ bits.

4.7.2. Firmas a ciegas definidas sobre puntos en una curva elíptica

La Tabla 4.7 compara dos esquemas de firmas a ciegas basadas en la criptografía sobre curvas elípticas, usando las curvas definidas en el campo binario 2^{251} bits. Como puede observarse, el esquema de Jena et al. calcula el mismo número de operaciones que el esquema de López et al. Note que, el comportamiento de los esquemas reporta que la primitiva de ocultamiento es la más costosa y la primitiva de firma es la más eficiente.

	Nuestra propuesta		Jena et al.	
	Multiplicación escalar	# Ciclos	Multiplicación escalar	# Ciclos
Oculta	3 mult. escalares	912,000	2 mult. escalares	608,000
Firma	1 mult. escalar	304,000	1 mult. escalar	304,000
Descubre	– –	–	– –	–
Verificación	2 mult. escalares	608,000	3 mult. escalares	912,000
Total	6	1,824,000	6	1,824,000

Tabla 4.7: Esquemas definidos sobre curvas elípticas binarias con $n = 251$ bits.

4.7.3. Esquemas definidos sobre emparejamientos bilineales

La firma a ciegas de Gao et al. por estar definida en la criptografía basada en la identidad, usa una autoridad generadora de llaves denominada PKG, mientras que la firma de Boldyreva usa una autoridad certificadora (AC) para avalar la relación entre la llave pública y el signatario de la firma.

Como puede observarse en la Tabla 4.8, la firma de Boldyreva es tres veces más eficiente que la firma de Gao et al., al utilizar el mínimo de operaciones criptográficas por primitiva, a comparación de la firma de Gao et al. la cual requiere calcular el doble de multiplicaciones escalares y de funciones de emparejamiento que la firma de Boldyreva.

	Boldyreva		Gao et al.	
	Operación	# Ciclos	Operación	# Ciclos
Oculta	1 mult. escalar 1 H_1	383,900 328,950	1 mult. escalar 1 H_1	383,900 328,950
Firma	1 mult. escalar	383,900	3 mult. escalares	1,151,700
Descubre	1 mult. escalar	383,900	3 mult. escalares 3 emparejamientos	1,151,700 7,470,000
Verificación	2 emparejamientos 1 H_1	4,980,000 328,950	4 emparejamientos 1 H_1	9,960,000 328,950
Total	–	6,789,600	–	20,775,200

Tabla 4.8: Esquemas de firma a ciegas definidos sobre emparejamientos asimétricos con $|r| = 254$-bits.

4.8. Sumario

En este capítulo se propuso un esquema de firma a ciegas basado en la criptografía sobre curvas elípticas. Nuestra propuesta cumple con los requerimientos de seguridad de los esquemas de firma a ciegas y es más eficiente que el esquema de Jena et al. el cual está definido también sobre curvas elípticas.

Además, se analizaron siete esquemas de firma a ciegas, que como en el capítulo anterior, nos permitió visualizar la eficiencia y la seguridad de cada uno.

Es notorio que la firma a ciegas de Boldyreva tiene ventaja sobre todas las firmas restantes. La firma producida es la más corta, el paso de mensajes es el mínimo requerido, ocupa únicamente un factor de opacidad y la implementación sobre emparejamientos simétricos resulta más eficiente. Además, cuenta con una demostración de seguridad que garantiza que es una firma resistente a la falsificación y a la vinculación entre el usuario y la firma solicitada. Por todas estas razones, seleccionamos la firma de Boldyreva para ser el bloque principal en nuestro esquema de votación electrónica.

Capítulo 5

Esquemas de votación electrónica

El avance de la tecnología ha permitido el desarrollo de diferentes aplicaciones que automatizan procesos los cuales se desarrollan manualmente. Ejemplo de esto son los procesos electorales los cuales pueden ser automatizados de diferentes formas. En este capítulo, presentamos los conceptos básicos, requerimientos, funcionalidad y seguridad de los sistemas de votación electrónica. Listamos algunos de los protocolos de seguridad propuestos en la literatura, dos de ellos que consideramos fueron la base de la funcionalidad para los protocolos publicados actualmente. Para finalizar, mencionamos algunos ejemplos de sistemas electrónicos de votación que han sido implementados.

5.1. Tipos de votaciones electrónicas

El uso de las comunicaciones y la tecnología en los procesos electorales tiene como objetivo proveer a los votantes la oportunidad de emitir su voto electrónicamente. Por un lado, la telefonía fija o móvil y el Internet son algunos de los medios usados para transmitir el voto electrónico y por otro, los dispositivos electrónicos más comunes para emitirlo son las computadoras personales (PC), máquinas de registro directo (DRE), tarjetas electrónicas, teléfonos celulares, entre otros.

Los sistemas de votación electrónica ofrecen una forma rápida y cómoda de manifestar el voto, sin embargo, estos factores no implican la confianza en el proceso electoral. Desde las primeras propuestas de votaciones electrónicas, éstas han sido objeto de controversia, dado que los dispositivos electrónicos permiten la pérdida del control sobre una parte importante del proceso de votación: *la contabilidad de los votos* [4]. Motivo suficiente para que los sistemas de votación electrónica tengan que ser sujetos a un nivel de escrutinio más elevado.

Los dispositivos más comunes para obtener el voto en los sistemas de votación electrónica son las máquinas DRE (*Direct Recording Electronic*) con botones o pantalla táctil para emitir el voto; impresoras WPAT (*Voter Verified Paper Audit Trail*) para obtener el registro impreso de la emisión del voto; boletas PEB (P*ersonalized Electronic*

Ballot) usadas por las autoridades electorales para generar las boletas; tarjetas CFC (*Compact Flash Card*) usadas por los votantes para abrir y cerrar sus boletas electrónicas y dispositivos digitales como computadoras personales, teléfonos celulares, iPods, etc. conectados a una red pública o privada [92]. En este trabajo, estamos interesados en los sistemas de votación electrónica, los cuales utilizan la red Internet para transmitir los votos.

Independientemente del medio o dispositivo electrónico que se use para una votación, poner en marcha una elección electrónica pública engloba la creación de leyes electorales que definan con precisión el funcionamiento del sistema y los procedimientos a seguir en caso de cualquier contingencia. Para tal efecto, consideremos los requerimientos necesarios de un sistema de votación electrónica.

5.2. Requerimientos

Las votaciones electrónicas son más vulnerables que las convencionales debido al procesamiento digital de los datos electorales que pueden ser fácilmente manipulados. La meta es ofrecer a los votantes un ambiente electrónico de votación seguro con un bajo costo y mucho más eficiente. Sin embargo, la desconfianza por las tentativas de fraude a partir del uso de medios de comunicación y dispositivos electrónicos es una limitación para la puesta en práctica de las elecciones en su versión electrónica.

Las votaciones electrónicas no pueden emular en su totalidad a las convencionales, no obstante, hacen uso de herramientas criptográficas para satisfacer los requerimientos necesarios en una elección. Los requerimientos pueden dividirse en tres categorías: generales, específicos y electorales [75].

Los requerimientos electorales son los que deben cubrirse de acuerdo al tipo de elección y leyes electorales. Establecen los periodos de tiempo para cada proceso en la elección como son el periodo de registro de los votantes, la votación y de conteo de los votos. Este tipo de requerimientos varían dependiendo del tipo de elección, es decir, si es local, estatal o nacional, en caso de candidatos para algún puesto de gobierno, o si es para alguna empresa o escuela.

Los requerimientos específicos son los particulares para cada tipo de sistema, por ejemplo, si es *multi-usuario* donde varios votantes pueden votar simultáneamente o *multi-electoral* si varias elecciones se desarrollan al mismo tiempo.

Los requerimientos de seguridad son necesarios para protegerse de potenciales ataques provenientes de la red. Así, se sugieren en [65] los siguientes criterios:

- ▷ *Autenticación*: sólo los votantes incluidos en el padrón electoral serán autorizados para emitir su voto.
- ▷ *Unicidad*: ningún votante debe votar más de una sola vez.
- ▷ *Exactitud*: el sistema debe registrar y procesar los votos correctamente.
- ▷ *Integridad*: los votos no deben ser modificados, olvidados, borrados o no detecta-

dos.

- ▷ *Verificación y auditoría*: debe ser posible verificar que al final del proceso electoral, todos los votos fueron contados correctamente, demostrando así la honorabilidad del sistema.
- ▷ *Confiabilidad*: el sistema debe ser robusto, sin pérdida de votos, sin fallas tanto en los servidores como en la comunicación a través de Internet.
- ▷ *Anonimato y no Coerción*: nadie debe ser capaz de determinar el valor del voto ni de vincularlo con el votante.
- ▷ *Flexibilidad*: se debe contar con una variedad de formatos para las boletas de votación a fin de hacerlos compatibles con las diferentes plataformas y tecnologías.
- ▷ *Conveniencia*: los votantes deben ser capaces de depositar su voto de manera rápida y con un mínimo de conocimientos informáticos.
- ▷ *Certificación*: el sistema electoral debe ser estable; en el caso de una elección oficial debe contar con los criterios necesarios para su validez obedeciendo la legislación vigente.
- ▷ *Transparencia*: los votantes deben entender y conocer el proceso de votación.
- ▷ *Costo-Eficiencia*: el sistema de votación debe ser computacionalmente eficiente.

5.3. Protocolos de seguridad

El uso de Internet como medio de comunicación para llevar a cabo un sistema de votación ha posibilitado nuevas formas de fraude electoral debido a los posibles ataques de seguridad inherentes a dicha red. Ello implica que en los sistemas de votación por Internet se requieran protocolos de seguridad que permitan garantizar dos objetivos en primera instancia contradictorios entre sí, los cuales consisten en la autenticación del votante por un lado, pero sin por esto vulnerar su derecho al voto secreto, por el otro.

En la literatura se han propuesto una variedad de protocolos criptográficos con la finalidad de evitar el fraude electoral y satisfacer el anonimato del votante. Grosso modo, estos protocolos pueden ser divididos en dos clases principales: basados en firmas a ciegas y basados en funciones homomórficas [89]. Sin embargo, podemos considerar también protocolos que utilizan redes mezcladas (mixnets) [46, 51] o pruebas de conocimiento nulo [25].

En los esquemas de votación, las funciones homomórficas proveen una herramienta para obtener el conteo de los votos sin descifrarlos [76], algunos de los esquemas propuestos basados en funciones homomórficas pueden ser consultados en [69, 71, 72].

Las firmas a ciegas en los esquemas de votación electrónica permiten la obtención del voto, sin identificar al votante que lo ha emitido.

A pesar de los protocolos propuestos de votaciones electrónicas usando tanto funciones homomórficas como firmas a ciegas, en ninguna de las dos clases se ha ofrecido una solución completa [89].

Debido a la sencillez y eficiencia con la que puede capturarse el voto, en esta tesis, nos enfocamos en los esquemas de votación electrónica basados en firmas a ciegas.

5.4. Protocolos de votación electrónica basados en firmas a ciegas

Desde la propuesta de Chaum hasta la fecha, muchos protocolos de votación electrónica han sido propuestos usando firmas a ciegas.

La Figura 5.1 muestra la funcionalidad de este tipo de esquemas. En general, se requiere como mínimo tres entidades (dos electorales y el votante) y por lo menos tres fases importantes la autenticación del votante, la votación y el conteo de los votos.

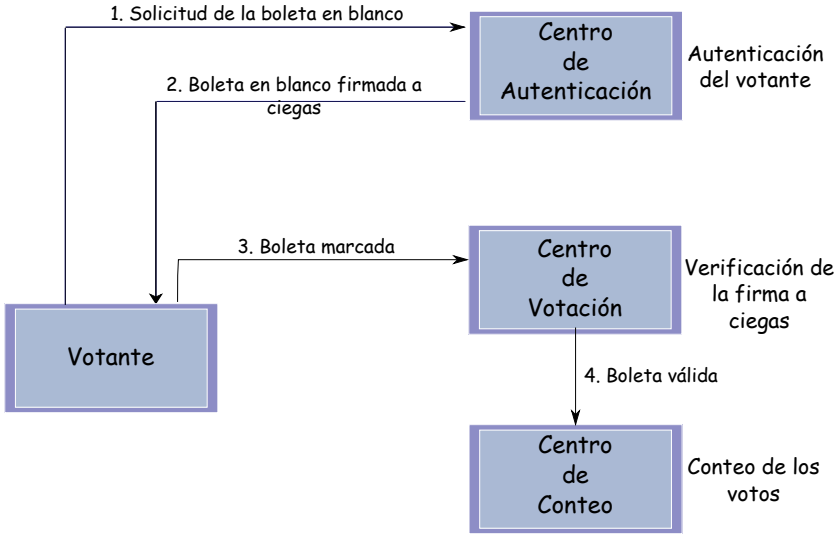

Figura 5.1: Funcionalidad de los esquemas basados en firmas a ciegas.

En 1992, Fijioka, Okamoto y Ohta [31] propusieron un esquema de votaciones electrónicas de gran escala. En 1997, Cranor y Cytron [24] presentaron una propuesta basada en la de Fujioka et al., denominada *Sensus*. En el 2005, Baiardi et al.[3] mostraron que Sensus permitía a las autoridades electorales hacerse pasar por los votantes que se abstenían de votar para ellos emitir el voto y propusieron un nuevo esquema llamado *Seas*. Finalmente, Chou et al. [22] indicó que Seas era un protocolo ineficiente y presentaron una versión corregida basada en emparejamientos bilineales, sin usar firmas a ciegas.

A continuación se presentan tres esquemas que fueron publicados recientemente y un cuarto que ofrece una propiedad interesante como la identificación del votante malicioso que vota de forma fraudulenta.

5.4.1. Protocolo de Kharchineh y Ettelaee

Kharchineh y Ettelaee [48] propusieron un esquema que usa como herramienta criptográfica principal la firma digital RSA y una firma a ciegas propuesta por ellos mismos. Ambas firmas son implementadas sobre un grupo multiplicativo. El protocolo consiste de cinco fases establecidas entre el votante y cuatro entidades electorales.

La funcionalidad del esquema se muestra en la Figura 5.2.

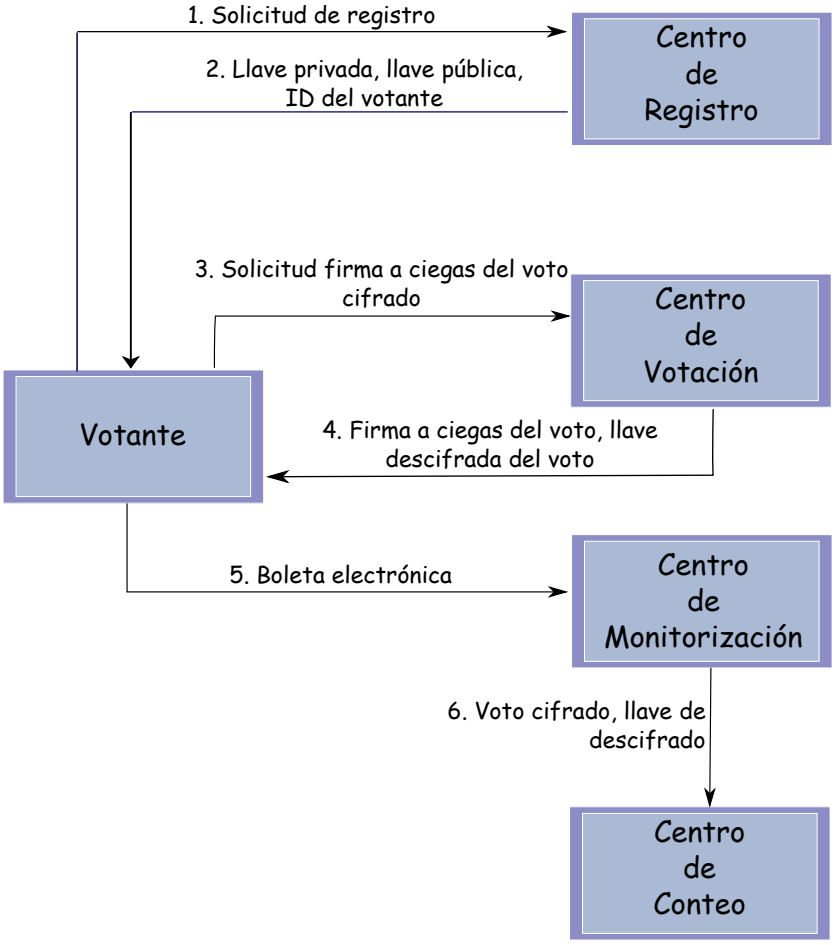

Figura 5.2: Esquema de votación electrónica de Kharchineh y Ettelaee

En una fase previa al protocolo, las entidades electorales intercambian sus llaves públicas $(PK_{CR}, PK_{CM}, PK_{CC})$ para el Centro de Registro (CR), el Centro de Monitorización (CM) y el Centro de Conteo (CC), respectivamente.

Fase de registro. cada votante solicita su registro enviando su ID al CR quien genera la llave privada SK_V, la llave pública PK_V y una clave de registro $h(ID)$ para cada

votante. Por último, el CR envía $(SK_V, PK_V, h(ID))$ al votante y $(ID, PK_V, h(ID))$ al Centro de Votación (CV).

Fase de votación. el votante, usando el par $(ID, h(ID))$, abre una sesión con el CV quien responde con la llave pública general para los votos PK_{voto}, la llave pública del Centro de Monitorización PK_{CM} y la lista m de los candidatos. De la lista, el votante elige el voto m', lo cifra con PK_{voto} y oculta a ciegas el mensaje utilizando los siguientes parámetros de dominio: el número primo p y un generador α de orden p.

$$C = PK_{voto}(m')$$
$$C' \equiv C + h \bmod (p-1)$$

donde h es el factor de opacidad y es elegido aleatoriamente en el rango $[2, p-1]$. El mensaje oculto C' es enviado al CV quien genera el par (t, t') para utilizarlo como acuse de recibo y produce la firma utilizando su llave privada a exclusiva para las firmas a ciegas.

$$t = H(ID||r)$$
$$t' = SK_{CR}(t)$$
$$\gamma \equiv \alpha^k \bmod p$$
$$\overline{\delta} \equiv (C' - (a+k))\gamma \bmod (p-1)$$

r es elegido aleatoriamente y k es la llave de sesión de la firma a ciegas. Finalmente, el CV cifra todos los valores con la llave pública del votante.

$$Z = PK_V(\gamma, \overline{\delta}, t', t)$$

El votante descifra el mensaje proveniente del CV, descubre la firma $\overline{\delta}$ y genera la boleta la cual es cifrada con la llave pública del CM.

$$(\gamma, \overline{\delta}, t', t) = SK_V(Z)$$
$$\delta \equiv \overline{\delta} - \gamma h \bmod (p-1)$$
$$X = PK_{CM}(C, \gamma, \delta, t', t)$$

Fase de monitorización. una vez recibida la boleta cifrada del votante, el CM la descifra y verifica la firma digital de t y la firma a ciegas del voto.

$$(C, \gamma, \delta, t', t) = SK_{CM}(X)$$
$$t' \stackrel{¿?}{=} SK_{CR}(t)$$
$$\alpha^{\gamma C} \stackrel{¿?}{=} \alpha^{\delta}(\beta\gamma)^{\gamma}$$

donde $\beta = \alpha^a \bmod p$. Si ambas firmas son correctas entonces el CM cifra, con la llave pública del CC, el par (C, t) que contiene el número de registro válido t del votante y el voto contenido en C.

$$W = PK_{CC}(C, t)$$

Fase de conteo y control. el CC recibe los votos cifrados y después de cerrada la fase de votación y monitorización, el CV envía la llave privada de los votos SK_{voto} al CV quien descifra todos los votos para su conteo.

$$(C, t) = SK_{CC}(W)$$
$$m' = SK_{voto}(C)$$

Finalmente el CC publica el par (t,contabilizado) para que el votante verifique a través de t si su voto fue contabilizado.

Eficiencia

El esquema propuesto por Kharchineh y Ettelaee, como se muestra en la Tabla 5.1 requiere de seis operaciones privadas RSA, cuatro públicas RSA y cuatro exponenciaciones.

Lo que implica que el esquema sea ineficiente si se requiere de un nivel de seguridad alto como por ejemplo de 128 bits, debido a que la operación privada de RSA es una exponenciación con un exponente del orden de 3072 bits.

Fase	Operación criptográfica		
	RSA-privada	RSA-pública	Exponenciación modular
Votación	2	3	1
Monitorización	2	1	3
Conteo	2	0	0
Total	6	4	4

Tabla 5.1: Número de operaciones criptográficas por fase, esquema de Kharchineh y Ettelaee.

Seguridad

Kharchineh y Ettelaee no hicieron el análisis de seguridad para su propuesta. No obstante, la *autenticación* se realiza a través de la sesión iniciada con el ID del votante al inicio de la fase de votación. La *integridad* se cumple gracias al cifrado del voto y de la boleta, utilizando RSA. La *unicidad* de las boletas se satisface al verificar que la información del acuse de recibo t sea única, ya que fue generado con el ID del votante y un número aleatorio r los cuales se suponen únicos. La *verificación* se realiza cuando el Centro de Conteo publica el par $(t, contabilizado)$, donde t es un acuse de recibo entregado al votante en el momento de aceptar como válida la boleta de electrónica. Por tanto, el votante tiene la opción de verificar que su boleta fue tomada en cuenta, pero no puede comprobar que el valor de su voto fue contabilizado correctamente. Por último el *anonimato*, propiedad que está condicionada al esquema de firma a ciegas que ellos mismos propusieron, es decir, si la firma a ciegas cumple con la propiedad de *no-vinculable* entonces el votante no podrá ser relacionado con su voto.

Vulnerabilidad

La autenticación se realiza con una sesión entre el votante y el centro de votación sin utilizar un esquema de firma digital que asegure la identidad del votante.

Independientemente del uso de la firma a ciegas para cumplir el anonimato del votante, el acuse de recibo t puede rastrearse para encontrar la identidad del votante. Dado que el Centro de Votación (CV) genera el valor aleatorio r y después calcula el par (t, t'),

$$t = H(ID||r)$$
$$t' = SK_{CR}(t)$$

Y envía la boleta de votación $X = PK_{CM}(C, \gamma, \delta, t', t)$ al Centro de Monitorización (CM). Nótese que t y t' es información disponible en claro en la boleta. Una vez que se verificó la validez de la boleta, ésta es enviada al Centro de Conteo (CC).

Por último, el CC publica la lista $(t_1, \textit{contado}), ..., (t_n, \textit{contado})$ de los n votos válidos recibidos, donde $t_i = h(ID_i||r_i)$ representa el acuse de recibo del i-ésimo votante.

Entonces, el CV puede rastrear la boleta en todas las fases del esquema, sólo haciendo una búsqueda coludida con el CM y el CC del valor t en las boletas.

5.4.2. Protocolo de Li et al.

El esquema propuesto por Li et. al. [53] utiliza tanto la firma digital como la firma a ciegas RSA. El protocolo consiste de cuatro fases, con un intercambio de nueve mensajes entre el votante y cinco entidades electorales. La funcionalidad del esquema se muestra en la Figura 5.3.

Fase de registro. una Autoridad Certificadora (AC) genera un par único de llaves (SK_{voto}, PK_{voto}) para cifrar y descifrar los votos. PK_{voto} es dada a cada votante y SK_{voto} es entregada al Centro de Conteo (CC) al finalizar el periodo de votación. Por su parte, cada votante solicita su registro enviando su ID a la AC quien genera el par (SK_V, PK_V) y envía la PK_V al Centro de Autenticación.

Fase de autenticación. el votante descarga la boleta en blanco M, la marca con el valor de su voto M' y la cifra utilizando la llave pública general de los votos PK_{voto}. Además, genera un nombre de usuario para iniciar sesión en la próxima fase utilizando su identificador ID y un valor aleatorio a. Entonces, oculta la cifra del voto y el nombre de usuario usando los factores de opacidad r_1 y r_2.

$$C = PK_{voto}(M')$$
$$ID_1 = H(ID||a)$$

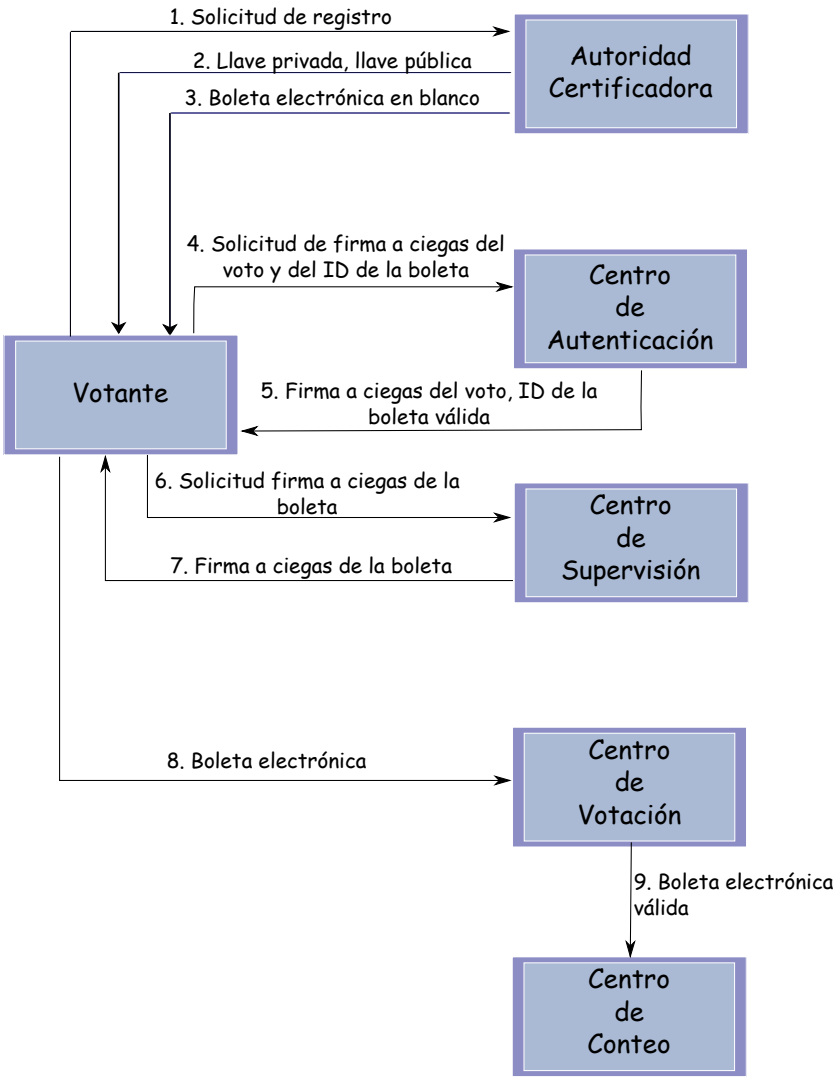

Figura 5.3: Esquema de votación electrónica de Li et al.

$$X = B(C, r_1)$$
$$Y = B(ID_1, r_2)$$

donde B es el algoritmo de ocultamiento de la firma a ciegas de Chaum. Con la finalidad de ser autenticado, el votante firma los mensajes ocultos y posteriormente los cifra con la llave pública del Centro de Autenticación (CA).

$$X_1 = SK_V(X)$$
$$Y_1 = SK_V(Y)$$
$$Z = PK_{CA}(X, X_1, Y, Y_1)$$

Una vez que el CA recibe Z, lo descifra, verifica las firmas X_1 y Y_1. Si son válidas entonces produce las firmas a ciegas, las cifra y se las envía al votante.

$$X \stackrel{¿?}{=} PK_V(X_1);\ Y \stackrel{¿?}{=} PK_V(Y_1)$$
$$X_2 = SK_{CA}(X);\ Y_2 = SK_{CA}(Y)$$
$$W = PK_V(X_2, Y_2)$$

El votante recibe W y descubre las firmas. Para asegurarse que el CA fue quien firmó, realiza la verificación de las firmas.

$$SG_1 = B^{-1}(X_2, r_1);\ SG_2 = B^{-1}(Y_2, r_2)$$
$$C \stackrel{¿?}{=} PK_{CA}(SG_1);\ H(ID||a) \stackrel{¿?}{=} PK_{CA}(SG_2)$$

Para obtener un acuse de recibo, el votante solicita una firma a ciegas de C, utilizando el factor de opacidad r_3, al Centro de Supervisión (CS) y cifra la boleta con la llave pública del CS.

$$X_3 = B(C, r_3)$$
$$Z_1 = PK_{CS}(X_3, H(ID||a), SG_2)$$

Z_1 es recibido por el CC quien lo descifra y verifica la firma SG_2. Si es válida entonces firma a ciegas y produce un acuse de recibo utilizando $H(ID||a)$ y un valor aleatorio b.

$$H(ID||a) \stackrel{¿?}{=} PK_{CA}(SG_2)$$
$$X_4 = SK_{CS}(X_3)$$
$$SG_3 = SK_{CS}(H(H(ID||a)||b))$$

El votante verifica la firma SG_3, descubre X_4 y verifica que después de retirar el factor de opacidad la firma haya sido producida por el CC.

$$SG_4 = B^{-1}(X_4, r_3)$$
$$H(H(ID||a)||b) \stackrel{¿?}{=} PK_{CS}(SG_4);$$

Fase de votación. el votante abre una sesión con el Centro de Conteo (CC) utilizando $H(ID||a)$ como nombre de usuario y SG_2 como contraseña. Para autorizar el acceso el CC verifica la firma SG_2 y si es válida entonces permite la recepción de la boleta $(SG_1, SG_3, H(H(ID||a)||b), SG_4)$ por parte del votante a través de un canal seguro y verifica las firmas.

$$C \stackrel{¿?}{=} PK_{CA}(SG_1)$$
$$C \stackrel{¿?}{=} PK_{CA}(SG_3)$$

Si son válidas entonces verifica el acuse de recibo.

$$H(H(ID||a)||b) \stackrel{¿?}{=} PK_{CS}(SG_4)$$

Fase de conteo. el Centro de Conteo (CC) recibe la llave privada SK_{voto} de la AC, descifra todos los votos y publica el acuse de recibo $H(H(ID||a)||b)$ con los resultados.

Eficiencia

El esquema propuesto por Li et al. utiliza únicamente operaciones privadas y públicas RSA. Como se observa en la Tabla 5.2 el esquema utiliza el doble de operaciones públicas con respecto a las privadas. Sin embargo, como en el caso anterior, este esquema puede ser ineficiente dado que requiere un número considerable de operaciones para la emisión del voto por votante.

Fase	Operación criptográfica	
	RSA-privada	RSA-pública
Autenticación	6	13
Votación	0	3
Conteo	1	0
Total	7	15

Tabla 5.2: Número de operaciones criptográficas por fase, esquema de Li et al.

Seguridad

Respecto a la seguridad, Li et al. presentaron un análisis de los siguientes requerimientos. La firma digital de Chaum permite la *autenticación* del votante. El cifrado del voto con RSA garantiza su *integridad*. La *unicidad* es posible gracias a la sesión establecida entre el votante y el Centro de Conteo utilizando el valor $H(ID||a)$ generada de forma única por votante.

La *verificación* se realiza cuando el Centro de Conteo publica los acuses de recibo $H(H(ID||a)||b)$, los cuales fueron producidos aleatoriamente, por tanto no es posible que las entidades electorales rastreen al votante y dado que el recibo no contiene el valor del voto, el votante tampoco puede comprobar por quién voto. Por último, el *anonimato* del votante se garantiza por la firma a ciegas de Chaum producida para el valor del voto y el acuse de recibo.

Vulnerabilidad

Para garantizar la seguridad del esquema, es necesario de cinco entidades electorales y de ocho interacciones entre el votante y las entidades electorales. La principal vulnerabilidad del esquema es la constante comunicación del votante con todas las entidades electorales, ya que requiere de ocho interacciones y un número considerable de operaciones criptográficas como se indicó en la Tabla 5.2.

5.4.3. Protocolo Chung y Wu

Chung y Wu [23] presentaron un protocolo donde se utiliza el esquema RSA como cifrador y como firma a ciegas. La propuesta consiste de cuatro fases, en cinco mensajes entre el votante y dos entidades electorales. La funcionalidad del esquema se muestra en la Figura 5.4.

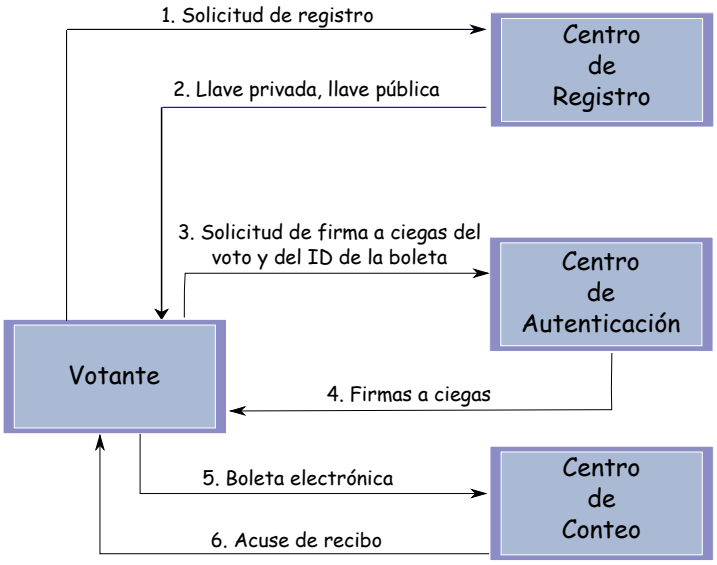

Figura 5.4: Esquema de votación electrónica de Chung y Wu.

Fase de iniciación. el votante obtiene de una Autoridad Certificadora (AC) una tarjeta inteligente y elige a su candidato M_d. La tarjeta contendrá el valor del voto M_d, y dos identificadores PW_n y PW_e que servirán para iniciar una sesión en el sistema. Además de la tarjeta, la AC proporciona al votante un par de llaves (SK_V, PK_V) y es generada también el par de llaves exclusiva para el cifrado del voto (SK_{voto}, PK_{voto}).

Fase de autenticación. la AC publica la lista de candidatos M. El votante inicia sesión usando PW_n, selecciona el número de su candidato y genera un número especial n_v que garantizará la validez de la boleta. Oculta ambos mensajes y los firma para enviarlos a la AC.

$$m' = B(m, r);\ n' = B(n', b)$$
$$s = SK_V(H(n_v))\ ;\ sn' = SK_V(sn')$$

La AC recibe las firmas y las verifica. Si ambas firmas son válidas entonces firma a ciegas y las envía al votante.

$$m' \stackrel{¿?}{=} PK_V(s);\ n' \stackrel{¿?}{=} PK_V(sn')$$
$$s' = SK_{AC}(m');\ n' = SK_{AC}(n')$$

Fase de votación. el votante descubre las firmas y construye la boleta que contiene dos números aleatorios r_1 y r_2, el voto m, la firma s, el número especial $H(n_v)$ y la firma sn.

$$s = B^{-1}(s', r);\ sn = B^{-1}(n', b)$$
$$Bs = (r_1||r_2||m||s||H(n_v)||sn)$$

El votante cifra la Bs con la llave PD_{boleta} y la envía al Centro de Conteo (CC).

Fase de publicación. el CC recibe la llave privada SK_{boleta} de la AC, descifra las boletas y verifica la firma de $H(n_v)$ y de m. Para cada boleta con firmas válidas, el CC verifica que no haya algún valor de $H(n_v)$ repetido para evitar el doble voto. Hecho lo anterior, para cada boleta se produce un acuse de recibo cifrando el valor r_1 con una llave simétrica s. Finalmente, el CC cuenta los valores de m y los publica. Si algún votante desea verificar que su voto fue contabilizado correctamente entonces debe acudir al CC para solicitar el descifrado de r_1 y confirme el valor de su voto.

Eficiencia

El esquema propuesto por Chung y Wu utiliza operaciones públicas y privadas RSA. La Tabla 5.3 presenta el número de operaciones criptográficas por fase, donde se requiere únicamente de siete operaciones privadas en contraste con el esquema de Li que requiere el doble.

Fase	Operación criptográfica	
	RSA-privada	RSA-pública
Autenticación	4	5
Votación	1	2
Publicación	0	0
Total	5	7

Tabla 5.3: Número de operaciones criptográficas por fase, esquema de Chung y Wu.

Seguridad

Chung y Wu presentaron un análisis de las propiedades que cumple su propuesta. La *autenticación* se cumple a través de la firma digital RSA que permite al Centro de Autenticación confirmar la identidad del votante. Como en el esquema anterior, la *integridad* del voto se garantiza gracias al cifrado de la boleta usando RSA. La *unicidad* se cumple gracias al número especial n_v el cual es único por votante y por tanto único por boleta. Para la *verificación*, el Centro de Conteo publica la lista de acuses de recibo de las boletas válidas contabilizadas, el votante entonces puede verificar si su acuse de recibo se encuentra contenido en dicha lista. Una opción más de verificación es posible,

si el votante acude al Centro de Conteo para solicitar el descifrado de su voto, pero tal acción se realiza de forma personal. El *anonimato* se cumple gracias a la firma a ciegas de Chaum producida para el voto, y el número especial n_v que garantiza la validez de la boleta.

Vulnerabilidad

En la fase de inicialización, una tarjeta inteligente es asignada a cada votante. La tarjeta contiene las llaves de sesión que permitirá al votante acceder a la boleta en la fase de votación. Además contiene el valor del voto deseado, el cual es utilizado para verificar, en la fase de publicación, si la boleta fue producida fraudulentamente. Sin embargo, con la finalidad de evitar el fraude electoral se pone en riesgo el anonimato del votante, puesto que la tarjeta es utilizada en todas las fases del esquema.

5.4.4. Protocolo de Mu y Varadharajan basado en emparejamientos bilineales

En 1998, Mu y Varadharajan propusieron un esquema de votación electrónica que además de proveer el anonimato del votante y la identificación de votos duplicados o falsos también identificaba al votante malicioso que los emitía. Herramientas criptográficas como la firma digital ElGamal [28], la firma a ciegas RSA [18], estampas de tiempo, certificados digitales para los votantes y los servidores y la concatenación de bits fueron usadas en las distintas fases del esquema.

Este esquema fue fuertemente atacado y varias versiones fueron propuestas para tratar de cubrir las debilidades encontradas. En 2003, Lin et al. en [54] indicaron que el esquema permitía la falsificación de las boletas, se podía rastrear la identidad del votante a partir de su boleta de votación, así como efectuar el doble voto y propusieron su propia versión, la cual corregía los problemas encontrados. En 2004, Yang et al. [95] propusieron un nuevo esquema que aseguraban era seguro contra los ataques mencionados. Hwang et al. en 2005 [42], propusieron un esquema basado en la versión de Lin et al. En 2007 Rodríguez et al. [78] demostraron que el uso de la firma ElGamal provocaba un problema funcional por la forma en como se generaba la boleta electrónica y propusieron modificar el esquema para usar la firma digital DSA en lugar de la de ElGamal.

En 2008, presentamos una versión de este esquema utilizando emparejamientos bilineales [55]. Para comprobar la funcionalidad del esquema propuesto realizamos su implementación en un sistema denominado SEVI [56].

En la Figura 5.5, se muestra la interacción entre el votante y los servidores electorales. En la fase inicial de autenticación, el SA recibe las solicitudes para el envío de la boleta en blanco únicamente a los votantes autenticados. Una vez recibida la boleta en blanco el votante emite el voto y envía la boleta marcada al SV, en la fase de votación. Por su parte, el SV verifica la validez de la boleta y la almacena. Finalmente, en la fase de conteo, el SV envía todas las boletas válidas al SC quien cuenta los votos y publica

los resultados.

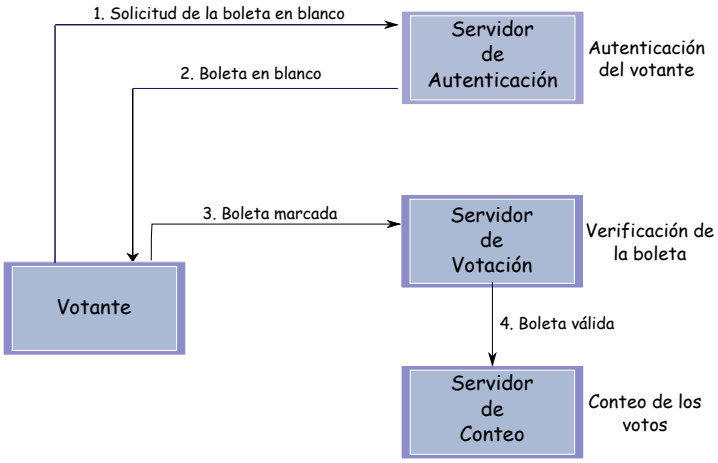

Figura 5.5: Estructura y funcionalidad del esquema de Mu y Varadharajan basado en emparejamientos.

La Tabla 5.4 presenta las cuatro fases de SEVI. En la primera de ellas, el votante es realmente un ciudadano que solicita el ingreso de su información a la lista nominal. Una vez que el Servidor de Registro (SR) ha verificado la información correspondiente, genera un par de llaves (privada,pública) para el ciudadano.

En la fase de autenticación, el ciudadano genera un par de llaves (d_{voto}, V_{voto}) que usará como seudónimo para firmar el voto y mantenerlo protegido de cualquier modificación. Para mantener un control en los seudónimos, el Servidor de Autenticación (SA) firmará únicamente un seudónimo por ciudadano. Con la finalidad de mantener oculta la relación entre el ciudadano y su boleta, la firma es producida a ciegas.

La firma digital ECDSA es utilizada para firmar el voto, por tanto, el ciudadano debe generar una llave de sesión k. Sin embargo, para que el esquema identifique al votante tramposo, k es calculada con la multiplicación $k = k_1 * k_2$, donde k_1 es generada por el votante y k_2 por el SA. Así, si el votante ha utilizado en más de una ocasión la misma llave de sesión, como se mostró en la Sección 3.5.3, es posible obtener k y con ello identificar al votante a través de k_2. Entonces, el ciudadano calcula el punto $R = k_1 k_2 P$ y su coordenada x_R será el valor r de la firma ECDSA. Finalmente, el ciudadano obtiene del SA, dos firmas a ciegas: una que avala la llave pública del seudónimo y otra que protege el punto R para evitar la relación entre el votante y el valor k_2 a través de la firma digital ECDSA.

En la fase de votación, el ciudadano ahora es votante, puesto que ya tiene en su poder toda la información necesaria para construir la boleta electrónica y emitir su voto. Hasta este punto, el votante ha generado la llave privada del seudónimo d_{voto}, ha calculado la llave de sesión $k = k_1 k_2$ y sólo le resta producir el valor s de la firma ECDSA del voto. Finalmente el votante envía la boleta electrónica al Servidor de Votación, la

cual consiste en dos firmas a ciegas $(\gamma_{V_{voto}}, \gamma_R)$, una firma digital (r, s) y sus respectivos mensajes $(V_{voto}, R, voto)$.

Recibida la boleta, el Servidor de Votación comprueba que el seudónimo haya sido autorizado y que el punto R haya sido calculado entre el votante y el SA, verificando las firmas a ciegas $\gamma_{V_{voto}}$ y γ_R con la llave pública del SA. Después, corrobora que el voto no haya sido modificado, verificando la firma digital (r, s) con la llave pública V_{voto}, donde r es tomado de la coordenada x del punto R convertida a un número entero módulo n. Si todas las firmas son válidas entonces la boleta es almacenada como válida y como falsa en caso contrario.

Al terminar el periodo de votación, el SV envía únicamente las boletas almacenadas como válidas al Servidor de Conteo (SC). Entonces, el SC realiza la búsqueda de boletas que contengan firmas iguales. Si encuentra una similitud en algún par entonces obtiene la llave de sesión k y solicita al SA la identidad del ciudadano tramposo. El protocolo entre las entidades se muestra en la Tabla 5.4

Vulnerabilidad

El esquema propuesto tiene la propiedad de identificar al votante tramposo a través de la firma digital ECDSA (r, s) donde r es la coordenada x del punto $R = k_1 k_2 P$, el cual está generado conjuntamente por el votante con el valor k_1 y por el SA con k_2, donde k_2 es el Id del votante. Si un votante utiliza en más de una ocasión a R entonces es posible obtener la llave de sesión $k_1 k_2$ según el ataque indicado en 3.5.3 e identificar al votante que emitió su voto dos veces.

Aunque tal identificación es atractiva en un esquema de votación electrónica, el ingreso de k_2 en la firma digital ECDSA provoca un problema mayor.

Con la finalidad de ocultar la relación del votante con el voto, el punto R de la firma digital es firmado a ciegas por el SA con el esquema de Boldyreva. Como se muestra en 4.4.5, la firma a ciegas requiere de la función H_1 para evitar la falsificación. Sin embargo, ingresar k_2 en el punto R significa calcular $\hat{S} = d_{SA} k_2 \hat{R} = d_{SA} b k_2 k_1 P$, donde b es el factor de opacidad. Si H_1 se utiliza entonces la ecuación seria $S = d_{SA} k_2 \hat{M} = k_2 b H_1(R')$, donde R' es el punto R convertido a cadena. Al final, para verificar la firma, el valor de k_2 no forma parte del argumento de H_1, lo que implica que vaya en claro en la boleta. No obstante, si H_1 no se utiliza, el votante puede crear una boleta falsa sin la intervención del SA y puede crear las firmas a ciegas tantas veces como lo desee y votar esa misma cantidad de veces sin ser identificado como fraudulento.

Consideramos que la característica de identificar al votante tramposo implica tener un esquema susceptible a ataques como es el doble voto o la relación entre el votante y su voto, por la creación de la llave de sesión $k = k_1 k_2$ entre el votante y el SA.

En el siguiente año, Asadpour y Jalili [1] demostraron que los esquemas basados en el de Mu y Varadharajan eran susceptibles al doble voto por la condición de la creación de la llave de sesión entre el votante y el SA.

Fase de registro

CIUDADANO — SA

d_V — d_{SA}

$V_V = d_V P$ — $V_{SA} = d_{SA} P$

Fase de autenticación

$d_{voto} \in [2, n-1]$

$V_{voto} = d_{voto} P$

$k_1 \in [2, n-1]$

$R = k_1 P$

$b \in [2, n-1]$

Ocultamiento de V_{voto}, R_1

$\tilde{V} = bV_{voto}$, $\tilde{R} = bR_1$

$$\{Cert_V, \tilde{V}, \tilde{R}, \{\tilde{V} \| \tilde{R} \| t\}_{d_V}\}$$

$$\longrightarrow$$

$k_2 \in [2, n-1]$

$\tilde{\gamma}_V = d_{SA}(\tilde{V} + \tilde{R})$

$\tilde{\gamma}_R = d_{SA}(k_2 \tilde{R})$

$$\{ENC(k_2||t)_V, \tilde{\gamma}_V, \tilde{\gamma}_R, \{\tilde{\gamma}_V \| \tilde{\gamma}_R \| t\}_{d_{SA}}\}$$

$$\longleftarrow$$

Descubrimiento de $\tilde{\gamma}_V$ y $\tilde{\gamma}_R$

$\gamma_{V_{voto}} = b^{-1} \tilde{\gamma}_V$

$\gamma_R = b^{-1} \tilde{\gamma}_R$

Fase de votación

VOTANTE — SV

Generación de la firma digital ECDSA

$R = k_2 k_1 P = (x_R, y_R)$

$s = (k_2 k_1)^{-1}(voto + x_R d_{voto}) \bmod n$

$$B = \{V_{voto}, R, s, voto, \gamma_{V_{voto}}, \gamma_R\}$$

$$\longrightarrow$$

$e(V_{SA}, V_{voto} + R) \stackrel{¿?}{=} e(P, \gamma_{V_{voto}})$

$e(V_{SA}, R) \stackrel{¿?}{=} e(P, \gamma_R)$

$ECDSA\_VER(V_{voto}, R, s, voto)$

Fase de Conteo

SC — SV

Comparación y contabilización — ⟵ Boletas válidas

Tabla 5.4: Esquema de Mu y Varadharajan basado en emparejamientos bilineales.

5.5. Sumario

En este capítulo, listamos las definiciones, propiedades y requerimientos de los esquemas de votación electrónica. Consideramos a los esquemas de votación electrónica basados en firmas a ciegas ya que permiten un protocolo entre los votantes y los servidores electorales con un mínimo de mensajes y sobretodo claridad en la funcionalidad.

Hicimos una revisión en la literatura sobre los esquemas propuestos usando firmas a ciegas como bloque principal y analizamos los esquemas de Kharchineh y Ettelaee, Li et al. y Chung y Wu publicados recientemente.
Los tres esquemas presentados siguen una misma estructura en el paso de mensajes y la generación de la boleta electrónica. Kharchineh y Ettelaee y Chung y Wu usan un par de llaves temporales para cifrar el voto, mientras que Li et al. utilizan las firmas digitales.

Todos los esquemas requieren de una fase de registro o inicialización antes de la captura del voto y todos requieren generar valores aleatorios para ser utilizados como seudónimos y producir un acuse de recibo que garantizará al votante la recepción de su boleta en el Centro de Conteo.

Propusimos también una modificación al esquema presentado por Mu y Varadharajan, el cual satisfacía la identificación del votante tramposo. Realizamos la comprobación de funcionalidad y eficiencia del esquema propuesto en una implementación sobre un sistema de votación electrónica denominado SEVI. Sin embargo, el análisis de seguridad permitió distinguir que identificar al votante tramposo a través de la firma digital ECDSA implicaba tener un esquema débil a ataques como el doble voto y concluimos que era necesario construir un esquema más robusto que tal vez no identifique a los votantes tramposos pero que sí obligatoriamente distinga boletas fraudulentas para evitar el doble voto.

Capítulo 6

Esquema seguro de votación electrónica propuesto

Conocidas las herramientas necesarias y el estado del arte en esquemas de votación electrónica, en este capítulo, presentaremos nuestra principal contribución: un esquema seguro de votación electrónica usando funciones de emparejamiento. Primero, daremos una breve explicación de la funcionalidad del esquema. Posteriormente, presentaremos el protocolo establecido entre las entidades y terminaremos con el análisis de seguridad y eficiencia del mismo.

6.1. Funcionalidad

Nuestro esquema está dividido en cuatro fases. Como se muestra en la Figura 6.1, la fase de registro, necesaria para la generación de la lista nominal, la fase de autenticación donde los votantes obtienen la boleta en blanco, la fase de votación donde se emite el voto y por último la fase de conteo para la obtención de los resultados. Cada fase consiste en el protocolo entre el votante y las entidades electorales, el cual se discute a continuación.

6.1.1. Fase de registro

En cualquier sistema de elecciones, la votación se desarrolla en base a una lista nominal que contiene a todos los integrantes con derecho a emitir el voto. Para la generación de la lista nominal, las autoridades electorales identifican plenamente al usuario o ciudadano e indiscutiblemente lo dotan con información única que les garantizará identificarlo al momento de la emisión del voto.

Para fines prácticos y de seguridad, nuestro esquema de votación electrónica utiliza una lista nominal que contiene tres categorías en la información del ciudadano: la *personal* como el nombre completo y fecha de nacimiento, la *electoral* que se refiere a la

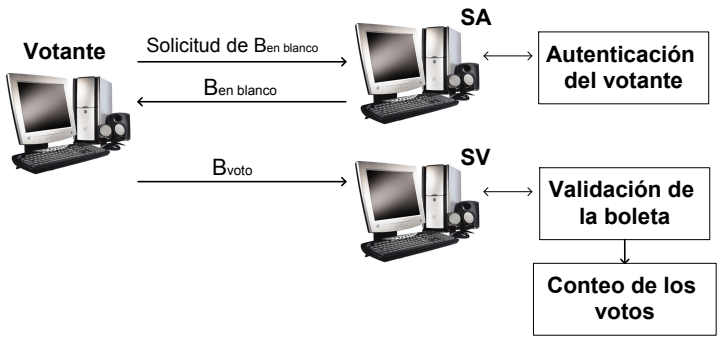

Figura 6.1: Fase de autenticación

localidad de vivienda, distrito electoral y zona estatal y por último, la información de *seguridad* que consiste en un certificado digital, el cual garantiza la relación entre el ciudadano y su llave pública.

La información personal y electoral es proporcionada por el ciudadano. Al contrario, la de seguridad es generada por una autoridad electoral. Tal entidad es realmente una *Autoridad Certificadora (AC)* que tiene varias tareas. La primera es comprobar que la información personal y electoral del ciudadano sea autentica. La segunda es generar el par de llaves privada y pública en conjunto con el ciudadano. La tercera y última es generar el certificado digital evidentemente de la llave pública del ciudadano.

Para evitar la usurpación de identidades por parte de los ciudadanos y garantizar que la generación del par de llaves criptográficas sea exclusiva para los ciudadanos válidos, consideramos la generación de la lista nominal de forma manual. Es decir, el ciudadano debe acudir a las Autoridades Certificadoras correspondientes, con la documentación necesaria que le acredite como válido para ser ingresado a la lista nominal.

Como en cualquier sistema electoral, el ciudadano es responsable de salvaguardar la información que le identificará como elector. De tal manera, que es de gran importancia que el ciudadano mantenga la llave privada en secreto para garantizar el buen funcionamiento del sistema de elecciones.

Al finalizar el periodo de registro, la Autoridad Certificadora entregará al Servidor de Autenticación (SA) la lista nominal electoral. Como es de esperarse, la AC genera un identificador único para el ciudadano al momento de registrarlo, tal información permitirá al SA ubicar el registro del ciudadano en la lista nominal y procederá a autenticarlo siguiendo los pasos indicados en la siguiente fase.

6.1.2. Fase de autenticación

El objetivo de esta fase es la generación de la boleta en blanco únicamente para los votantes válidos. Entonces, el votante solicita la boleta electoral al servidor de auten-

ticación (SA). El paso de mensajes entre el votante y el SA concluye con la boleta en blanco emitida por este último. La boleta en blanco consiste en una firma a ciegas. Lo siguiente que debemos preguntarnos es, cuál es el mensaje oculto que el SA firma. Para visualizar la respuesta analizaremos el comportamiento del votante.

Para garantizar la integridad del voto, el votante requiere producir una firma, sin embargo, si el votante usa su llave privada para generar la firma del voto, entonces su anonimato se verá comprometido porque es necesaria su llave pública para la verificación de la firma. La solución a este dilema, es darle la libertad al votante de generar un par de llaves temporales (d_t, V_t), donde la llave pública V_t no tenga alguna relación con el votante ofreciendo de esta manera el anonimato en la verificación de la firma.

Las llaves temporales, no obstante, nos traen un problema aún mayor, pues si el votante no es vigilado en la generación de las llaves temporales entonces, tiene total libertad de generar muchos pares de claves y emitir el mismo número de votos. Por tal razón, el SA tiene la tarea de limitar al votante a generar un solo par, produciendo una firma digital para la llave pública temporal, como una especie de certificado digital. Con la finalidad de evitar que la llave pública temporal sea vinculada con el votante, por el SA, la firma es producida a ciegas. En resumen, el votante genera el par (d_t, V_t) y toma la llave pública V_t como mensaje de la firma a ciegas. Una vez que el votante ha ocultado a V_t, envía la solicitud de la firma a ciegas al SA.

La solicitud contiene el identificador del elector denotado como ID_V, la llave pública temporal oculta $\tilde{V}_t$ y la firma digital $S_{\tilde{V}_t}$ generada con la llave privada del elector.

Por su parte, el SA al recibir la solicitud, realiza varias tareas antes de producir la firma a ciegas. Primero, verifica que el elector se encuentre registrado, usando el ID_V para ubicarlo en la lista nominal entregada por el AC. Segundo, autentica al votante verificando la firma digital $S_{\tilde{V}_t}$ utilizando la llave pública avalada por el certificado digital contenido en el registro del elector. Por último, revisa que no haya generado ya una firma a ciegas para ese solicitante. Una vez hechas todas las verificaciones, el SA produce la firma a ciegas $\tilde{S}$ y marca el registro del elector indicando la emisión de la boleta en blanco.

Al recibir la firma a ciegas, el elector la descubre, obtiene la firma S_{V_t} y se convierte en votante. La Figura 6.2 nos muestra el procedimiento de la fase de autenticación.

6.1.3. Fase de votación

En esta fase el votante construye y envía la boleta electrónica al Servidor de Votación (SV) quien es el encargado de recibirla, verificarla y almacenarla.

Hasta este punto, el votante tiene en su poder la firma a ciegas de la llave pública temporal, lo que significa que tiene el permiso del SA para generar una boleta electrónica. Por tanto, produce la firma digital S_V del valor de voto, utilizando la llave privada temporal d_t.

La boleta electoral electrónica denotada como B consiste de la llave pública temporal

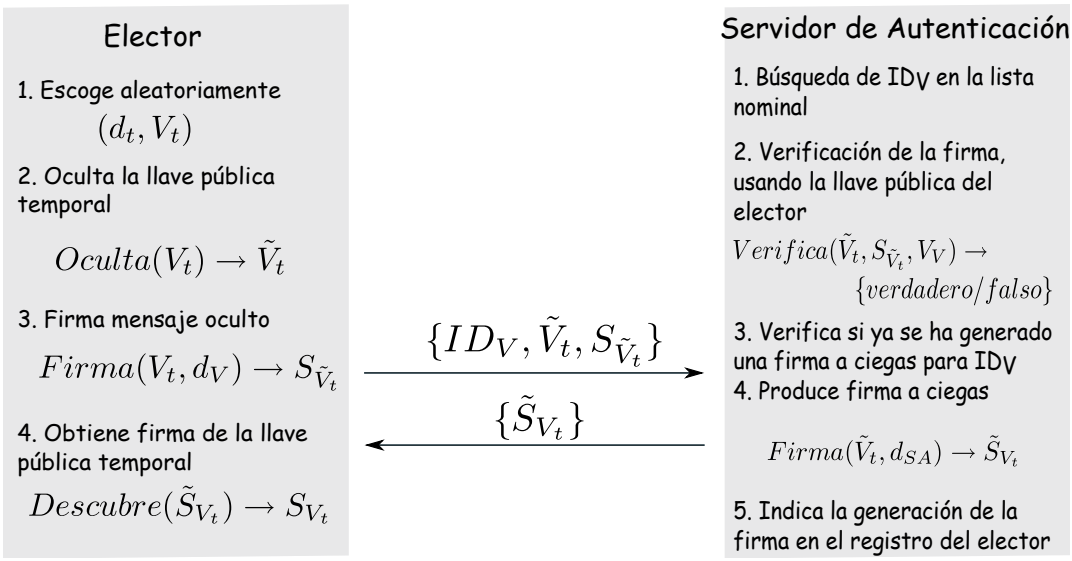

Figura 6.2: Fase de autenticación

V_t, su firma a ciegas S_{V_t}, el valor del voto v y su firma digital S_v. Entonces, B con sus cuatro valores está lista para ser enviada al SV.

Al recibir B, el SV verifica primero que la firma a ciegas haya sido emitida por el SA usando la llave pública de esta entidad, con lo cual se garantizaría la validez de la llave pública temporal V_t. Si la verificación es *verdadera*, concluye que el votante fue autenticado por el SA. El segundo paso es la verificación de firma S_v utilizando la llave pública temporal, para garantizar la integridad del voto.

Si ambas verificaciones arrojan un resultado *verdadero* entonces B es genuina, de lo contrario es inválida. Para las dos opciones, B es almacenada con el estatus correspondiente y es generado un acuse de recibido con toda la información contenida en la boleta y un valor aleatorio a que tiene como propósito evitar que el votante extraiga a través del acuse, el valor del voto. Por último, el acuse es enviado al votante. La Figura 6.3 indica los pasos que se efectúan en esta fase.

6.1.4. Fase de conteo

Una vez que el SV cierra el periodo de votación, antes de contabilizar los votos, el Servidor de Autenticación está obligado a publicar la lista de votantes a los que se les generó la boleta en blanco. Tal publicación tiene la finalidad de que el SA se comprometa a reportar todas las firmas a ciegas que generó y que éstas estén relacionadas con las firmas producidas por los votantes al realizar la solicitud.

Hecha la publicación del SA, el SV revisa si existen boletas duplicadas o posiblemente fraudulentas, analizando el contenido de cada boleta. Dado que las llaves temporales son

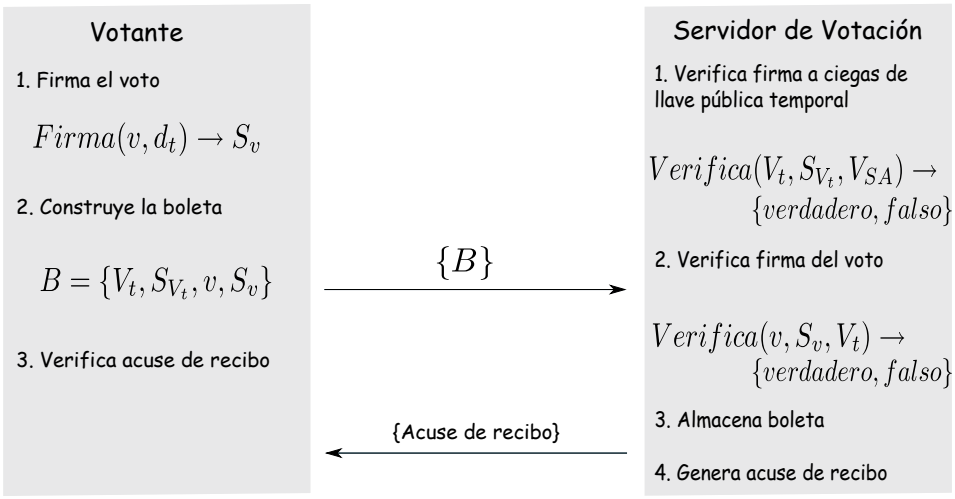

Figura 6.3: Fase de votación

generadas por cada votante en un espacio de búsqueda aproximadamente de $[1, r-1]$ con $|r| = 254$ bits, la probabilidad de que se elijan dos pares idénticos es desdeñable.

Lo anterior implica que las boletas contienen información que se presume única. Es decir, una llave pública temporal no repetida genera una firma a ciegas diferente para cada votante. De igual manera, una llave privada temporal única genera una firma del voto diferente a las demás firmas aun cuando el valor del voto sea el mismo.

En el caso de hallarse una firma o una llave pública temporal igual en dos o más boletas se concluiría que éstas fueron generadas de forma fraudulenta o que la boleta fue enviada en más de una ocasión. Para ambos casos, el SV toma la primera boleta como válida y marca las restantes como fraudulentas o inválidas.

Al finalizar la comprobación de todas las boletas almacenadas, el SV extrae el valor de los votos únicamente de las boletas válidas y realiza el conteo. La presentación de los resultados del proceso electoral va acompañada de la publicación de las listas de los acuses de recibo, tanto de las boletas válidas como de las marcadas como fraudulentas. Por tanto, los votantes pueden revisar en las listas si su boleta fue considerada en el conteo o no.

6.2. Protocolo de seguridad

Los pasos para producir una boleta electrónica se resumen en la generación de dos firmas digitales, las cuales, requieren de un mínimo de operaciones criptográficas. Los parámetros de dominio definidos y la notación utilizada en nuestro esquema son los siguientes.

Sean $\mathbb{G}_1$ y $\mathbb{G}_2$ dos grupos aditivos y $\mathbb{G}_T$ un grupo multiplicativo, todos de orden

primo r. La función de emparejamiento bilineal *ate* óptimo $a_{opt} : \mathbb{G}_2 \times \mathbb{G}_1 \rightarrow \mathbb{G}_T$ y la función *map-to-point* $H_1 : \{0,1\} \rightarrow \mathbb{G}_1$.

Notación

- $\diamond$ $\{d_{SA}, V_{SA}\}$: llaves privada y pública del Servidor de Autenticación.
- $\diamond$ $ID_V, \{d_V, V_V\}, Cert_V$: identificador, llave privada, llave pública y el certificado digital del votante.
- $\diamond$ $\{d_t, V_t\}$: llave privada temporal y llave pública temporal.
- $\diamond$ $x \| y$: denota los valores x y y concatenados.
- $\diamond$ $b \in \mathbb{Z}_r^*$: denota un factor de opacidad.
- $\diamond$ t: denota una estampa de tiempo.
- $\diamond$ $\{m\}_X$: denota la firma corta del mensaje m generado por la entidad X.
- $\diamond$ H: función picadillo.

6.2.1. Generación de llaves

El par de llaves criptográficas de los votantes y de los servidores son generadas en la fase de registro y a través de la Autoridad Certificadora.

Los servidores publican sus llaves para la verificación de las firmas, mientras que los votantes no tienen necesidad de enviar alguna información pública a los servidores debido a que la AC envía la lista nominal que contiene los certificados digitales de los hasta en ese momento electores al Servidor de Autenticación.

6.2.2. Fase de autenticación

Como primer paso, el elector selecciona aleatoriamente la llave privada temporal d_t en un rango de $[1, r-1]$ y calcula la llave pública temporal V_t. Ambas llaves son el *seudónimo* del votante.

$$\begin{array}{lcl} Q & \in & \mathbb{G}_2 \\ d_t & \xleftarrow{\$} & \mathbb{Z}_r^* \\ V_t & = & d_t Q \in \mathbb{G}_2 \end{array}$$

Como el mensaje de la firma a ciegas es V_t el cual es un punto en $\mathbb{G}_2$, el elector debe convertirlo a una cadena que permita la utilización de la función H_1. Recordemos que la función *map-to-point* es determinante en los esquemas basados en emparejamientos ya que sin ella, es posible la falsificación de la firma. Entonces, utilizando una función *map-to-string* denotada como $m2s : \mathbb{G}_2 \rightarrow \{0,1\}^{4|r|}$, el elector convierte a V_t en una

cadena m misma que es convertida a un punto nuevamente usando H_1.

$$\begin{aligned} V_t &= d_t Q \\ m &= m2s(V_t) \\ M &= H_1(m) \in \mathbb{G}_1 \end{aligned}$$

Con el mensaje listo, el elector lo oculta según lo indica la firma a ciegas de Boldyreva, utilizando un factor de opacidad b y calculando una multiplicación escalar en $\mathbb{G}_1$.

$$\begin{aligned} b &\xleftarrow{\$} \mathbb{Z}_r^* \\ \tilde{M} &= bM \in \mathbb{G}_1 \end{aligned}$$

Para garantizar su identidad, el elector firma el mensaje oculto $\tilde{M}$ utilizando su llave privada d_V.

$$S_{\tilde{M}} = d_V \tilde{M}$$

La solicitud se compone de un mensaje que contiene el ID_V del votante, una estampa de tiempo t para evitar ataques de repetición, el mensaje oculto $\tilde{M}$ y la firma $S_{\tilde{M}}$.

$$\{ID_V, t, \tilde{M}, S_{\tilde{M}}\}$$

Por su parte, el SA recibe la solicitud, busca al elector en la base de datos a través del ID_V. Si lo encuentra, toma la llave pública y verifica la firma del mensaje oculto.

$$\begin{aligned} a_{opt}(Q, S_{\tilde{M}}) &= a_{opt}(Q, d_V \tilde{M}) \\ &= a_{opt}(d_V Q, \tilde{M}) \\ &= a_{opt}(V_V, \tilde{M}) \end{aligned}$$

Si la igualdad se satisface, entonces el SA firma el mensaje oculto y lo envía al elector junto con la estampa de tiempo.

$$\tilde{S} = d_{SA} \tilde{M} \in \mathbb{G}_1$$

$$\{t, \tilde{S}\}$$

Para terminar con la fase de autenticación, el elector descubre la firma a ciegas usando el factor de opacidad b.

$$S_{V_t} = b^{-1} \tilde{S}$$

6.2.3. Fase de votación

Ya que se tiene la firma a ciegas que se puede traducir como una boleta en blanco, para formar la boleta electrónica, el votante firma el voto utilizando la firma corta de Boneh y la llave privada temporal d_t.

$$\begin{aligned} v &= voto \\ S_v &= d_t H_1(v) \in \mathbb{G}_1 \end{aligned}$$

construye la boleta B y la envía al Servidor de Votación.

$$B = \{V_t, S_{V_t}, v, S_v\}$$

Al recibir la boleta, el SV verifica la firma a ciegas de Boldyreva, para lo cual, debe obtener la cadena proveniente del punto V_t y utiliza la función H_1.

$$\begin{aligned} m &= m2s(V_t) \in \{0,1\}^{4|r|} \\ M &= H_1(m) \in \mathbb{G}_1 \end{aligned}$$

$$\begin{aligned} a_{opt}(Q, S_{V_t}) &= a_{opt}(Q, d_{SA}M) \\ &= a_{opt}(d_{SA}Q, M) \\ &= a_{opt}(V_{SA}, M) \end{aligned}$$

Si es válida, procede a verificar la firma corta de Boneh.

$$\begin{aligned} a_{opt}(Q, S_v) &= a_{opt}(Q, d_t H_1(v)) \\ &= a_{opt}(d_t Q, H_1(v)) \\ &= a_{opt}(V_t, H_1(v)) \end{aligned}$$

Si la verificación arroja *verdadero*, la boleta es almacenada como válida, en caso contrario como inválida.

Para terminar con la fase de votación, el SV genera un acuse de recibo con toda la información de la boleta. Específicamente, toma todos los valores de la boleta electrónica y los concatena junto con un valor aleatorio a para evitar la coerción de los votos, produce un digesto con una función picadillo estándar y lo firma para comprobar la legitimidad de la recepción de la boleta por parte del SV.

$$\begin{aligned} a &\xleftarrow{\$} \mathbb{Z}_r^* \\ ACK &= H(V_t||S_{V_t}||v||S_v||a) \\ S_{ACK} &= d_{SV} H_1(ACK) \end{aligned}$$

El votante recibe el acuse de recibo $ACK \in \mathbb{Z}_r^*$ y verifica la firma $S_{ACK} \in \mathbb{G}_1$.

$$\begin{aligned} a_{opt}(Q, S_R) &= a_{opt}(Q, d_{SV} H_1(ACK)) \\ &= a_{opt}(d_{SV} Q, H_1(ACK)) \\ &= a_{opt}(V_{SV}, H_1(ACK)) \end{aligned}$$

El votante conserva el acuse de recibo y la fase de votación entre él y el SV finaliza.

La Tabla 6.1 nos muestra el flujo de comunicación entre el elector/votante y los servidores electorales.

6.3. Análisis de seguridad

6.3.1. Requerimientos

Nuestro esquema satisface los requisitos más importantes del voto electrónico de la siguiente manera.

Anonimato del votante

En nuestro esquema se satisface el anonimato gracias a la generación del seudónimo, es decir, de las llaves privada y pública temporales, las cuales no tienen ninguna relación con el votante, debido a que d_t se genera de manera aleatoria.

$$\begin{aligned} d_t &\xleftarrow{\$} \mathbb{Z}_r \\ V_t &= d_t Q \in \mathbb{G}_2 \end{aligned}$$

Para controlar la generación de los seudónimos, el SA debe firmar la llave pública temporal V_t utilizando en conjunto con el votante la firma a ciegas de Boldyreva. El SA produce la firma del mensaje $\tilde{M}$ que contiene el valor de V_t convertido a cadena. Para conseguir m el SA requiere resolver la ecuación $\tilde{M} = bH_1(m)$, obtener b y posteriormente, encontrar la preimagen de $H_1(m)$.

$$\begin{aligned} m &= m2s(V_t) \in \{0,1\}^{4|r|} \\ \tilde{M} &= bH_1(m) \end{aligned}$$

Si H_1 es resistente a la preimagen y el PLDCE es difícil en el grupo $\mathbb{G}_2$, entonces en nuestra propuesta se satisface el anonimato del votante.

Por otra parte, después de ser enviada la boleta al SV, nuestro esquema garantiza que el SA no puede vincular al votante con su boleta gracias a la firma a ciegas de Boldyreva utilizada para firmar la llave pública temporal.

Elector		Servidor de Autenticación (SA)
$d_V \in \mathbb{Z}_r$		$d_{SA} \in \mathbb{Z}_r$
$V_V = d_V Q \in \mathbb{G}_2$		$V_{SA} = d_{SA} Q \in \mathbb{G}_2$
	Fase de autenticación	
$b, d_t \in \mathbb{Z}_r$		
$V_t = d_t Q \in \mathbb{G}_2$		
$m = m2s(V_t) \in \{0,1\}^{4\|r\|}$		
$\tilde{M} = bH_1(m)$		
$S_{\tilde{M}} = d_V \tilde{M}$		
	$\{ID_V, t, \tilde{M}, S_{\tilde{M}}\}$ $\longrightarrow$	
		$a_{opt}(Q, S_{\tilde{M}}) \stackrel{¿?}{=} a_{opt}(V_V, \tilde{M})$
		$\tilde{S} = d_{SA}\tilde{M}$
	$\{t, \tilde{S}\}$ $\longleftarrow$	
$S_{V_t} = b^{-1}\tilde{S}$		
	Fase de votación	
Votante		Servidor de Votación (SV)
$S_v = d_t H_1(v)$		
$B = \{V_t, S_{V_t}, v, S_v\}$	$\stackrel{\{B\}}{\longrightarrow}$	$m = m2s(V_t)$
		$a_{opt}(Q, S_{V_t}) \stackrel{¿?}{=} a_{opt}(V_{SA}, H_1(m))$
		$a_{opt}(Q, S_v) \stackrel{¿?}{=} a_{opt}(V_t, H_1(v))$
		$a \in \mathbb{Z}_r$
		$ACK = H(V_t\|\|S_{V_t}\|\|v\|\|S_v\|\|a)$
		$S_{ACK} = d_{SV} H_1(ACK)$
	$\{ACK, S_{ACK}\}$ $\longleftarrow$	
$a_{opt}(Q, S_{ACK}) \stackrel{¿?}{=} a_{opt}(V_{SV}, H_1(ACK))$		

Tabla 6.1: Protocolo seguro de votación electrónica basado en emparejamientos

En la ecuación $\tilde{M} = bH_1(m)$ se oculta la llave pública temporal, donde m es V_t convertido a cadena a través de la función $m2s$ y después de varias verificaciones el SA firma a ciegas $\tilde{S} = d_{SA}\tilde{M}$. Cuando el SV recibe la boleta, la llave temporal es conocida. Si ambos servidores unen sus fuerzas para rastrear a los votantes, entonces consiguen las siguientes tuplas,

Para un $V_{ti} = d_{ti}Q$, el SA necesita encontrar un b' tal que la siguiente ecuación se

SV	SA
$V_{t_1}, S_{V_{t_1}}$	$\tilde{M}_1, \tilde{S}_1$
$V_{t_2}, S_{V_{t_2}}$	$\tilde{M}_2, \tilde{S}_2$
$\vdots$	$\vdots$
$V_{t_n}, S_{V_{t_n}}$	$\tilde{M}_n, \tilde{S}_n$

satisfaga.

$$\tilde{M}_j \stackrel{¿?}{=} b' H_1(m2s(V_{ti}))$$

Lo que implica resolver el PLDCE en $\mathbb{G}_2$. Así, las tuplas cumplen con la propiedad de *no-vinculable* requerida en los esquemas de firmas a ciegas y por tanto los servidores están imposibilitados de obtener una relación entre el votante y su boleta electrónica, durante y después de la elección.

Exactitud y Unicidad

La exactitud y unicidad son requerimientos que exigen considerar únicamente una boleta por votante al momento de realizar la contabilización de los votos. Los resultados publicados después del proceso electoral deben concordar con el número de votantes que solicitaron la firma y con el número de votos contados, de tal manera que no existan más votos emitidos que boletas solicitadas. Para cubrir ambos requerimientos, nuestro esquema realiza varias tareas distribuidas en las tres fases.

En la fase de registro, la autoridad certificadora entrega al SA la lista nominal que contiene el ID_V y la llave pública del votante. Cuando recibe la solicitud de una boleta en blanco,

$$\{ID_V, t, \tilde{M}, S_{\tilde{M}}\}$$

busca a ID_V en la lista nominal, si lo encuentra, verifica que no haya sido generada ya una firma a ciegas para ese votante. Sino es así entonces autentica al votante verificando la firma con la llave pública contenida en la lista nominal para ID_V.

$$a_{opt}(Q, S) \stackrel{¿?}{=} a_{opt}(V_V, \tilde{M})$$

Si es válida entonces el SA produce la firma a ciegas y marca el registro del votante, indicando que la boleta en blanco ya fue generada, para evitar que se genere más de una boleta para el mismo votante.

En la fase de votación, el SV revisa la validez de las boletas verificando las firmas contenidas en ellas. Si la verificación arrojó verdadero para ambas firmas entonces la boleta es almacenada como válida, en caso contrario como es almacenada como inválida.

Una vez que el periodo de votación se cierra, el SV considera únicamente las boletas válidas comparando cada una de ellas con las demás para localizar alguna duplicidad

Boleta electrónica B
$\downarrow$
verificación de las firmas
$\swarrow \quad \searrow$
Boleta válida Boleta inválida

ya sea de forma accidental o intencional.

Cada boleta esta construida con dos firmas y dos mensajes. La firma a ciegas tiene como mensaje al punto V_t convertido a cadena con la función $m2s$. El punto V_t es la multiplicación escalar de d_t con el punto base $Q \in \mathbb{G}_2$. d_t es seleccionado aleatoriamente de $\mathbb{Z}_r^*$ con una probabilidad de $1/2^r$, lo que significa una probabilidad desdeñable que dos votantes elijan el mismo valor para d_t. Por tanto, la firma a ciegas será distinta para

Mensaje; Firma
$V_t = d_t Q$; S_{V_t}

Selección aleatoria de d_t en $\mathbb{Z}_r^*$
$d_t \xleftarrow{\$} \mathbb{Z}_r^*$

cada votante.

En la fase de votación, el votante utiliza la firma corta de Boneh para garantizar la integridad del voto. Aunque el valor del voto sea el mismo para muchos votantes, la firma se produce con la llave privada temporal d_t,

$$S_v = d_t H_1(voto)$$

lo que implica que todas las firmas sean distintas puesto que d_t tiene un valor distinto para cada votante. De esta manera, cada boleta está construida con un par de firmas únicas por votante.

Es importante notar que las firmas están relacionadas entre sí a través del valor d_t, (en la firma a ciegas con $V_t = d_t Q$ y en la firma corta con $S_v = d_t H_1(v)$). Por tanto, para identificar una boleta fraudulenta es suficiente con encontrar un par que contenga el mismo V_t.

$$\begin{aligned} B_1 &= \{V_{t1}, S_{V_t1}, v_1, S_{v1}\} \\ &\vdots \\ B_i &= \{V_{ti}, S_{V_ti}, v_i, S_{vi}\} \\ &\vdots \\ B_n &= \{V_{tn}, S_{V_tn}, v_n, S_{vn}\} \end{aligned}$$

Si un par de boletas son comparadas y tienen el valor V_t igual, entonces el SV considera a la segunda boleta recibida como fraudulenta y a la primera como genuina, tomando el voto de esta última para ser contabilizado.

Nuestro esquema cumple con la unicidad al generar no más de una boleta en blanco por votante y garantiza la exactitud de los resultados al contar únicamente un voto por votante.

Integridad

La firma producida para el voto, como todo esquema de firma digital, es sobre el digesto del mensaje. En el caso de la firma corta, es sobre el punto que proyecta la función H_1 de acuerdo al valor del voto v.

$$S_v = d_t H_1(v)$$

H_1 garantiza que cualquier modificación al voto resulte un punto distinto al punto proyectado con el valor original. Por tanto, al verificar la firma, se garantiza que el valor del voto no ha sido modificado en la transmisión de la boleta electoral hacia el servidor de votación.

No coerción

El votante puede mostrar a cualquier persona que emitió el voto por un candidato especial, pero no puede probar por quién voto cuando los resultados son publicados por el SV. El acuse de recibo,

$$\begin{gathered} a \xleftarrow{\$} \mathbb{Z}_r^* \\ ACK = H(V_t||S_{V_t}||v||S_v||a) \end{gathered}$$

es un digesto generado con la información de la boleta electrónica, incluido el voto y un valor aleatorio a, este último impide que el votante construya un acuse de recibo con su boleta electrónica y genere un digesto igual a alguno de los acuses publicados en la lista de boletas genuinas. Para generar un acuse de recibo igual al publicado por el SV, el votante necesita encontrar el valor aleatorio a en el intervalo $[1, r-1]$, con $|r| = 254$ bits.

Verificación y auditoria

El votante j puede verificar si el acuse de recibo entregado por el SV, se encuentra en la lista de boletas genuinas (válidas) o en la lista de boletas inválidas y confirmar si su voto fue contabilizado o no.

La verificación puede realizarse a distintos niveles. En nuestro caso, nos limitamos a indicarle al votante que su boleta fue válida o inválida a través del acuse de recibo ACK.

Al final del conteo, el SV publica una lista que contiene el acuse de recibo entregado a cada votante y la indicación de si la boleta fue válida o inválida. De tal manera, que el votante i al publicarse la lista realice una búsqueda de su acuse ACK_i y se informe si el voto fue contabilizado en caso de que resultara válida la boleta o ignorado en caso contrario.

Dada la información que almacenan los servidores de autenticación y de votación, la única posibilidad de rastrear el valor del voto para un votante determinado, es que el propio votante revele su identidad al indicar el valor de la llave publica temporal y el factor de opacidad empleado para ocultar a V_t, con lo cual, el SA puede relacionar el mensaje oculto $\tilde{M}$ con V_t. Por su parte, el SV a través de V_t identifica la boleta correspondiente y obtiene el valor del voto.

6.3.2. Ataques

Los puntos anteriores son respecto al cumplimiento de los requerimientos de seguridad de un esquema de seguridad para votaciones electrónicas. Lo siguiente se refiere a las herramientas que tenemos para resistir los ataques comunes a la implementación de nuestro esquema de seguridad.

Doble voto

Como se mostró en el requerimiento de unicidad, existen varios mecanismos para detectar las boletas fraudulentas como es la identificación de llaves temporales y/o firmas iguales. Sin embargo, algún votante malicioso puede intentar crear una boleta falsa con o sin la intervención del SA.

El primer caso se refiere a generar una boleta falsa a partir de la información producida entre el votante y el SA. Es decir, el votante ha generado una boleta en blanco en conjunto con el SA y tratará de producir una nueva boleta modificando la llave pública temporal.

Debido a que la firma a ciegas está definida con parámetros que permiten una seguridad de 128 bits, el votante no está habilitado para producir una firma falsa, por tanto, buscará romper la función $m2s : \mathbb{G}_2 \rightarrow \{0,1\}^{4|r|}$.

Dado que el grupo $\mathbb{G}_2$ es un subgrupo de $\tilde{E}(\mathbb{F}_{p^2})$, para todo punto $S = (x, y) \in \mathbb{G}_2$, las coordenadas x y y son de la forma $a_1\alpha + a_0$ con $a_1, a_0 \in \mathbb{F}_p$, lo que implica una longitud en bits de aproximadamente $2|r|$ para cada coordenada.

La función $m2s$ es una conversión de un punto en $\mathbb{G}_2$ a una cadena de longitud $4|r|$ de la forma $(a_{1x}||a_{0x}||a_{1y}||a_{0y})$ considerando la concatenación de la representación en bits de los coeficientes a_{1x}, a_{0x}, a_{1y}, a_{0y} de las coordenadas x y y, respectivamente. Lo que significa que la función $m2s$ tiene una correspondencia uno a uno entre un punto en $\mathbb{G}_2$ y una cadena $\{0,1\}^{4|r|}$.

El votante al producir una llave pública temporal genuina utilizando $m2s$ queda imposibilitado de usar un nuevo V_t puesto que la función $m2s$ garantiza que cualquier modificación en su argumento producirá una nueva cadena en su salida.

En el segundo caso, donde se intenta falsificar la boleta sin intervención del SA. El votante tiene una tarea más ardua, ya que requiere falsificar, para cualquier punto V_t, la firma a ciegas de Boldyreva la cual tiene como supuestos de seguridad al PLDCE y el PCDH en los grupos $\mathbb{G}_1$ y $\mathbb{G}_2$, agregándole también que está demostrada como segura bajo el modelo del oráculo aleatorio y está definida con parámetros de seguridad de 128 bits. Por tanto, el votante está imposibilitado para falsificar la firma a ciegas de Boldyreva.

Servidores maliciosos

Los servidores mantienen comunicación entre ellos para realizar la auditoria del sistema. El SV contiene la información de las boletas emitidas y el SA contiene la información de los votantes autenticados. De los servidores, el SA tiene la libertad de generar boletas y emitir votos.

La manera de evitar la usurpación por parte del SA es verificando las firmas cortas de los votantes enviadas al SA en el momento de solicitar la boleta en blanco. En la fase de autenticación el votante produce el mensaje oculto de V_t y lo firma,

$$\begin{aligned} d_t, b &\in \mathbb{Z}_r^* \\ V_t &= d_t Q \in \mathbb{G}_2 \\ m &= m2s(V_t) \\ \tilde{M} &= bH_1(m) \\ S &= d_V \tilde{M} \end{aligned}$$

Para falsificar S, el SA puede utilizar la información contenida en la lista nominal generada por la autoridad certificadora en la fase de registro. La lista contiene el ID, la llave pública V_V y el certificado digital $Cert_V$ de cada votante. Con toda esa información

el SA elige un registro y calcula lo siguiente:

$$\begin{aligned} b &\in \mathbb{Z}_r^* \\ S_{falsa} &= bV_V = bd_V Q \\ M_{falso} &= bQ \end{aligned}$$

para que S_{falsa} sea una firma válida se requiere que el mensaje sea $M_{falso} = bQ$ y con esto la verificación arroje verdadero. Dado que $bQ \in \mathbb{G}_2$ y la firma a falsificar se encuentra definida en el grupo $\mathbb{G}_1$, si el SA pretende crear una firma usando la llave pública de algún votante, tomando como mensaje el punto $M_{falso} \in \mathbb{G}_2$ entonces necesita alguna transformación para trasladar los puntos S_{falsa} y M_{falso} del grupo $\mathbb{G}_2$ al grupo $\mathbb{G}_1$ sin alterar la llave privada del votante para garantizar que la verificación sea válida.

Afortunadamente, los grupos $\mathbb{G}_1$ y $\mathbb{G}_2$ están definidos en distintas curvas de tal manera que sean linealmente independientes, propiedad que no permite el traslado de un grupo a otro. Por tanto, no es útil realizar una falsificación utilizando la llave pública del votante.

Sin la llave pública del votante, si el SA pretende usurpar la identidad de algún votante tendrá que crear una firma S que verifique como válida, lo que se considera una tarea intratable, dado que es necesario falsificar la firma corta, la cual, ha sido demostrada por sus autores como segura en contra del ataque de falsificación bajo el modelo del oráculo aleatorio.

No obstante, si se diera el caso de que el SA inventara firmas y que éstas no coincidieran con la llave pública contenida en el certificado digital del votante, los escrutadores, quienes realizan las auditorías, pueden rastrear la boleta falsa cuestionando al SA sobre la llave pública temporal inventada para ser identificada en alguna de las boletas que están almacenadas con el SV y con eso restar el valor del voto en los resultados finales.

Para evitar que el SA niegue la creación de boletas falsas, antes de realizar el conteo de los votos, el SA está obligado a publicar la lista de los votantes autenticados. Así, si un elector no emitió su voto podrá denunciar el fraude ante los escrutadores.

En el caso del SV, para que pueda generar una boleta falsa, requiere de la generación de la firma a ciegas producida por el SA. No obstante, el SV se enfrenta con el problema de obtener la firma del SA para la llave pública temporal que el mismo ha generado. Como estamos en el escenario en que ambos servidores son maliciosos, basta con que el SA produzca la firma de todas las llaves públicas temporales que el SV le solicite. Sin embargo, el SA cae en el problema expuesto anteriormente, donde en la auditoría los escrutadores pueden identificar si la firma a ciegas fue producida para un votante legítimo o imaginario.

Votantes maliciosos

La complicidad de votantes maliciosos puede derivar en la generación de boletas falsas sin ser detectadas por el SV. No obstante, la estructura de la boleta dificulta la

tarea de los votantes, si el objetivo es mezclar dos boletas genuinas B_1 y B_2 para generar una tercera falsa B_3.

$$\begin{aligned} B_1 &= \{V_{t_1} = d_{t_1}Q, S_{V_{t_1}}, v, S_{v_1} = d_{t_1}H_1(v)\} \\ B_2 &= \{V_{t_2} = d_{t_2}Q, S_{V_{t_2}}, v, S_{v_2} = d_{t_2}H_1(v)\} \\ &\Downarrow \\ B_3 &= \{V_{t_1} = d_{t_1}Q, S_{V_{t_1}}, v, S_{v_2} = d_{t_2}H_1(v)\} \end{aligned}$$

Dado que las firmas están vinculadas con el par de llaves temporal, tomar $S_{V_{t_1}}$ de B_1 y S_{v_2} de B_2 se traduce en la llave pública temporal V_{t_1} y la llave privada temporal d_{t_2}. Para que el ataque surta el efecto deseado, los votantes maliciosos necesitan encontrar una entero x en el intervalo $[1, r-1]$ que cumpla la siguiente igualdad $V_{t_1} = xd_{t_2}Q$, lo cual es el problema del logaritmo discreto en curvas elípticas.

6.4. Análisis de eficiencia

La Tabla 6.2 muestra el número de operaciones que calculan el votante y el servidor de autenticación en la fase de autenticación. De igual manera la Tabla 6.3 presenta las operaciones necesarias para garantizar una boleta correctamente construida por parte del votante y su verificación por parte del SV en la fase de votación.

Como puede observarse, se necesitan cinco multiplicaciones escalares y una función *map-to-point* para producir la boleta electoral. Con la finalidad de autenticar al votante, el SA requiere calcular dos emparejamientos bilineales.

En la fase de votación, el votante calcula una multiplicación escalar, dos funciones *map-to-point* y dos emparejamientos, mientras que el servidor de votación lleva la parte más complicada al calcular cuatro emparejamientos y dos funciones *map-to-point* para la verificación de las firmas contenidas en la boleta.

	Multiplicación escalar	Emparejamiento bilineal	Función especial *map-to-point*
Votante	4	0	1
SA	1	2	0

Tabla 6.2: Número de operaciones criptográficas en la fase de autenticación

Para aprovechar el uso de los emparejamientos y la elegancia de la firma corta al producir firmas más pequeñas que los demás esquemas de firma digital, se decidió utilizar la firma corta para autenticar al votante y al SV en las fases de autenticación y votación respectivamente. Sin embargo, es posible únicamente para esos dos caso utilizar cualquier esquema de firma digital. En contraste con la firma a ciegas de la llave pública temporal

	Multiplicación escalar	Emparejamiento bilineal	Función especial *map-to-point*
Votante	1	2	2
SV	1	4	3

Tabla 6.3: Número de operaciones criptográficas en la fase de votación

V_t y de la firma corta del voto v, las cuales para brindar la seguridad, mencionada en la sección anterior, es necesario que se conserven en el protocolo propuesto.

6.5. Comparación

En esta sección presentamos la comparación general y un análisis de eficiencia entre nuestra propuesta y tres esquemas que fueron publicados recientemente en la literatura y fueron abordados en la Sección 5.4.

6.5.1. Comparación general y de seguridad

La Tabla 6.7 muestra las características generales de los esquemas de votación electrónica, como son, el paso de mensajes entre entidades, el número de fases y el número de entidades. Es notorio que todos los esquemas requieren como mínimo cuatro fases para completar el proceso de votación para un votante, a excepción de Kharchineh y Ettelace [48] que utiliza una fase adicional.

Respecto a las entidades, la segunda propuesta necesita cinco autoridades electorales para interactuar con el votante, razón por la cual, son necesarios nueve mensajes para completar el proceso de votación para un solo votante. Nuestra propuesta junto con la de Chung y Wu [23] consideran el mínimo de fases, entidades y mensajes para producir una boleta y contabilizar el voto para un votante.

Esquema	# Fases	# Entidades	# Mensajes
Kharchineh y Ettelace	5	5	6
Li et al.	4	6	9
Chung y Wu	4	4	6
Nuestro esquema	4	4	6

Tabla 6.4: Comparación general entre los esquemas de votación electrónica.

Como puede observarse en la Tabla 6.5, las boletas producidas por los dos primeros esquemas son demasiado extensas al considerar por lo menos cuatro valores con una longitud de 3072 bits. En contraste con el esquema de Chung y Wu que requiere de una sola firma de 3072 bits de longitud. Aunque nuestra propuesta requiere de dos firmas para construir la boleta, la extensión en bits que ocupa es menor a la Chung y Wu, al necesitar aproximadamente 1500 bits. Es decir, la longitud de la boleta es menor que la longitud de una firma digital RSA con un nivel de seguridad de 112 bits que considera un módulo $|N| = 2048$ bits.

Esquema	Boleta producida	Longitud en bits
Kharchineh y Ettelace	$(r_1 \| \| r_2 \| \| m \| \| s \| \| H(n_v) \| \| sn)$	(3072,3072,2,3072,256,3072)
Li et al.	$(SG_1, SG_3, H(x), SG_4)$	(3072,3072,256,3072)
Chung y Wu	$PK_{CM}(C, \gamma, \delta, t', t)$	3072
Nuestro esquema	$(V_t, S_{V_t}, voto, S_{voto})$	((1016),2,(254,1),(254,1))

Tabla 6.5: Boletas producidas de los esquemas de votación electrónica.

La Tabla 6.6 muestra que nuestro esquema y el de Li et al [53] satisfacen la mayoría de los requerimientos de los esquemas de votación electrónica, a excepción del esquema de Kharchineh y Ettelace y de la propuesta de Chung y Wu donde es posible rastrear al votante a partir de su boleta electrónica.

Requerimientos	Kharchineh y Ettelace	Li et al.	Chung y Wu	Nuestro esquema
Autenticación	×	✓	✓	✓
Anonimato	×	✓	×	✓
Integridad	✓	✓	✓	✓
Exactitud	✓	✓	✓	✓
Auditoría	✓	✓	✓	✓
Unicidad	✓	✓	✓	✓

Tabla 6.6: Comparación general entre los esquemas de votación electrónica

Esquema	Herramientas criptográficas	Supuesto de seguridad	Longitud de los parámetros
Kharchineh y Ettelace	Firma digital RSA Firma a ciegas	PFI PLD	$\|n\|$ = 3072-bit $\|p\|$ = 3072-bit $\|q\|$ = 256-bit
Li et al.	Firma digital Firma a ciegas	PFI PFI	$\|n\|$ = 3072-bit $\|n\|$ = 3072-bit
Chung y Wu	Firma digital Firma a ciegas	PFI PFI	$\|n\|$ = 3072-bit $\|n\|$ = 3072-bit
Nuestra propuesta	Firma corta Firma a ciegas	DCHP DCHP	$\|r\|$= 254-bit, $k = 12$ $\|r\|$= 254-bit, $k = 12$

Tabla 6.7: Comparación de seguridad con un nivel de 128 bits.

6.5.2. Comparación de eficiencia

Consideramos la implementación de todos los esquemas con un nivel de seguridad de aproximadamente 128 bits. La tabla 6.7 nos muestra las herramientas criptográficas que cada esquema utiliza y la longitud en bits necesarios para el nivel de seguridad mencionado. La implementación fue desarrollada en un procesador Intel Core 2 Quad @2.4Ghz y los algoritmos utilizados para cada operación criptográfica se muestran en el Capítulo 2.

Para el esquema de Kharchineh y Ettelace se utiliza la firma digital RSA y una firma a ciegas propuesta por los mismos autores, basada en DSA. Por tanto, los primos p y q para la firma a ciegas deben ser de 3072 y 256 bits respectivamente, para alcanzar la seguridad de 128 bits. Los dos esquemas siguientes usan tanto firmas a ciegas como firmas digitales RSA con modulo de 3072 bits.

Nuestro esquema utiliza la aritmética de curvas elípticas, para una curva definida en un campo primo $\mathbb{Z}_p$ con un grado de encajamiento $k = 12$ y orden r. La longitud de r para obtener el mismo nivel que los esquemas anteriores es de aproximadamente 254 bits, lo que significa una gran reducción en la longitud de los mensajes entre entidades, de las firmas digitales y a ciegas, de la boleta electrónica y sobre todo del costo de implementación de las operaciones criptográficas.

La Tabla 6.8 presenta la eficiencia de cada esquema de acuerdo al número de ciclos que fueron obtenidos respecto al grupo abeliano donde las operaciones criptográficas

fueron definidas para un nivel de seguridad de 128 bits. La segunda columna nos muestra el número de operaciones públicas y privadas basadas en RSA para los primeros tres esquemas y el número de operaciones definidas en los grupos que nuestro esquema utiliza. Claramente el esquema de Chung y Wu es más eficiente que los esquemas de Kharchineh y Li. Por otra parte, nuestro esquema requiere de un número menor de ciclos a pesar de que el esquema de Chung tiene un número mínimo de operaciones criptográficas. Por tanto, concluimos que nuestro esquema es el más eficiente.

Esquema	# Operación criptográfica	# Ciclos
Kharchineh	4 RSA-publica	7,290,128
y Ettelace	6 RSA-privada	308,013,756
	4 DLP-exponenciaciones	87,135,920
		Total 402,439,804
Li et al.	15 RSA-publica	27,337,980
	9 RSA-privada	462,020,634
		Total 489,358,614
Chung y Wu	5 RSA-publica	9,112,660
	4 RSA-privada	205,342,504
		Total 214,455,164
Nuestra propuesta	1 multiplicación escalar en $\mathbb{G}_2$	1,027,400
	6 multiplicaciones escalares en $\mathbb{G}_1$	2,303,400
	6 H_1	1,973,700
	8 emparejamientos bilineales	19,920,000
	Total	**25,224,500**

Tabla 6.8: Comparación de eficiencia entre los esquemas.

6.6. Sumario

En este capítulo presentamos la principal contribución de este trabajo de tesis. Un esquema seguro para votaciones electrónicas que consiste de una interacción mínima entre el votante y dos servidores electorales.

Para conseguir mayor control en la generación de la lista nominal, nuestra propuesta requiere que la fase de registro se realice manualmente. De tal manera, que se le solicita al votante acudir a las instalaciones electorales con la finalidad de entregar la documentación necesaria para la generación de un certificado digital, el cual avalará su identidad ante los servidores electorales en las fases posteriores.

Realizamos también, el análisis de seguridad y una comparación de eficiencia de nuestra propuesta contra tres esquemas propuestos recientemente en la literatura, concluyendo que nuestro esquema es el más seguro y eficiente.

Capítulo 7

Conclusiones y trabajo futuro

7.1. Sumario

En esta tesis, presentamos un esquema de votación electrónica basado en firmas a ciegas. Nuestra propuesta tiene como bloques criptográficos principales los esquemas de firma digital y firma a ciegas.

Para seleccionar el primero de ellos, realizamos un análisis de las firmas digitales propuestas, desde las fundamentales como son RSA y DSA hasta las firmas definidas sobre la joven criptografía basada en la identidad con emparejamientos bilineales.

Respecto al segundo, nos dimos a la tarea de analizar siete esquemas de firma a ciegas, definidos en diferentes grupos ya sean aditivos o multiplicativos. Comenzamos con la primera firma a ciegas propuesta por Chaum basada en RSA y terminamos con un esquema utilizando emparejamientos bilineales y basado en la identidad.

Con la finalidad de hacer la comparación de seguridad de todos los esquemas, sumamos a nuestra investigación el tema de *seguridad demostrable* en los esquemas de firma digital. Lo que nos permitió valorar la seguridad tanto de los esquemas de firma digital como los de firma a ciegas.

La comparación de eficiencia, la realizamos en base a los grupos donde se encuentra definido cada esquema. Tomamos tres grupos especiales: *multiplicativos* con operaciones sobre enteros; *aditivos* con operaciones sobre puntos de una curva elíptica; y la combinación de ambos a través de la función de emparejamiento. Para presentar una comparación justa de los esquemas agregamos un capítulo con los fundamentos matemáticos y los algoritmos que se utilizaron para la implementación de cada operación criptográfica básica usada en las primitivas de los cripto-esquemas de llave pública presentados a lo largo de este documento, con un nivel de seguridad de 128 bits.

En el ámbito de las votaciones electrónicas, realizamos una investigación acerca de los esquemas de seguridad propuestos y los tipos principales existentes, como son los basados en funciones homomórficas y los basados en firmas a ciegas. Para garantizar una dirección correcta en la construcción de un nuevo esquema, revisamos la debilidades y

las fortalezas de cada tipo, considerando que las firmas a ciegas son una herramienta útil y eficiente que permite la creación de un esquema de votación electrónica con un mínimo número de mensajes, con alta seguridad, eficiencia en la implementación y sobretodo claridad en la funcionalidad.

Finalmente, presentamos nuestro esquema de seguridad para votaciones electrónicas basado en firmas a ciegas utilizando emparejamientos bilineales. La propuesta utiliza un par de llaves temporales que le permiten al votante firmar el voto para garantizar la integridad del mismo, utilizando la llave privada temporal para producir la firma corta y tomando la llave pública temporal para su verificación.

La llave pública está avalada por la firma a ciegas del Servidor de Autenticación y la llave privada es utilizada para producir la firma corta del voto. Ambas firmas usan la criptografía basada en emparejamientos, razón por la cual, la boleta electrónica es construida utilizando la arimética de curvas elípticas y es verificada usando la función de emparejamiento bilineal.

7.2. Conclusiones

Previo a esta tesis, los esquemas de seguridad para votaciones electrónicas basados en firmas a ciegas utilizando emparejamientos bilineales no habían sido explorados todavía. Por tanto, nuestra contribución principal es un esquema de seguridad novedoso, que gracias a que los emparejamientos bilineales fue posible ofrecer una eficiencia significativa en comparación con tres esquemas de votación electrónica propuestos recientemente.

La combinación de la firma a ciegas de Boldyreva y la firma corta de Boneh, ambas basadas en emparejamientos bilineales, permitió cumplir con los requerimientos principales de los esquemas de votación electrónica como son autenticación, unicidad, anonimato, no coerción, integridad, verificación y auditoria, colocando a nuestra propuesta como un esquema seguro de votación electrónica.

El uso de la firma a ciegas y la creación de llaves temporales como un seudónimo permitió satisfacer el anonimato del votante y evitar la coerción de votos. Por su parte, con la firma corta fue posible proteger la integridad del voto y autenticar al votante.

El análisis de seguridad y de eficiencia realizado para varios esquemas tanto de firma digital como de firma a ciegas, permitió identificar las ventajas y desventajas de cada uno.

Considerando la eficiencia de la firma ECDSA, se propuso un esquema de votación electrónica que combina la criptografía sobre curvas elípticas y la criptografía basada en emparejamientos. La propuesta es una versión mejorada del esquema de Mu y Varadharajan el cual pretendía detectar al votante tramposo. Sin embargo, el análisis de seguridad del esquema propuesto permitió distinguir que la llave de sesión generada por la firma digital abría la posibilidad de detectar al votante tramposo y al mismo tiempo de permitir el doble voto y la pérdida del anonimato del votante. Concluyendo que, a

pesar de que la firma ECDSA y las firmas a ciegas definidas sobre curvas elípticas son las mejores opciones en eficiencia para utilizarse en un esquema de votación electrónica, el principal inconveniente es el uso de la llave de sesión que como se explica en el Capítulo 5, abre la posibilidad de crear boletas falsas, en su aplicación a las votaciones electrónicas.

Considerando tres esquemas de votación electrónica propuestos recientemente en la literatura, realizamos una comparación general, de seguridad y eficiencia entre éstos esquemas y nuestra propuesta. Nuestro esquema realiza un número mínimo de operaciones criptográficas en un número menor de interacciones para la generación y verificación de la boleta y la contabilización de los votos que los demás esquemas.

Respecto a la seguridad de nuestra propuesta, dado que los esquemas de firma fueron definidos sobre grupos aditivos, es decir utilizando puntos sobre una curva elíptica, fue suficiente con definir la curva elíptica sobre un campo finito $\mathbb{F}_p$ con $|p| = 254$ bits, para ofrecer un nivel de seguridad de 128 bits en los esquemas de firmas y producir con eso una boleta electrónica con una longitud en bits mucho más pequeña que el resto de los esquemas.

Además, obtuvimos una ventaja significativa en la eficiencia, debido a la implementación de la aritmética en curvas elípticas y el uso de la función de emparejamiento más eficiente propuesta en la literatura. Con lo que, nuestro esquema de seguridad para votaciones electrónicas ofrece una opción segura, eficaz y eficiente para ser implementado en un sistema electrónico de votación.

Finalmente, muchos esquemas de votación electrónica han sido publicados en la literatura, no obstante, nuestra propuesta es el primer esquema que desarrolla un protocolo entre las entidades electorales y el votante totalmente basado en emparejamientos bilineales a través de la firma a ciegas de Boldyreva [11] y de la firma corta de Boneh et al. [12], permitiendo el paso de mensajes de longitud de aproximadamente 1500 bits, ofreciendo un nivel de seguridad de 128 bits con una longitud para la llave privada de 254 bits y obteniendo una eficiencia mucho mayor a comparación de los demás esquemas basados en firmas a ciegas que no utilizan emparejamientos bilineales.

7.3. Trabajo futuro

En nuestro esquema de votación, se satisfacen los requerimientos que consideramos más importantes en los esquemas de votación electrónica. Sin embargo, contemplamos los siguientes puntos como trabajo futuro.

1. Con la finalidad de demostrar la seguridad del esquema propuesto, tomamos como trabajo futuro la demostración formal de cada uno de los requerimientos que nuestro esquema cumple.

2. Siguiendo la línea de la seguridad en las votaciones electrónicas, no se descarta

la posibilidad de construir un esquema híbrido que utilice tanto funciones homomórficas como firmas a ciegas y redes mezcladas, con el objetivo de reducir las vulnerabilidades u omisiones que pueden presentar cada uno por separado.

3. Tomando en consideración que el requerimiento de verificación puede satisfacerse a distintos niveles, una garantía de confianza para el votante es cumplir este requerimiento de tal manera que le sea posible al votante obtener un acuse de recibo que le permita identificar si el valor de su voto fue sumado al resultado de la elección, sin perder el anonimato y sin dar pie a la coerción.

4. Como se mencionó en el Capitulo 5, existen distintas opciones para implementar elecciones electrónicas, entre ellas se encuentran los sistemas que utilizan urnas electrónicas. Una tarea importante sería considerar los protocolos de seguridad de las urnas electrónicas y hacer una comparación de costo-beneficio con respecto a los sistemas implementados en línea.

5. Por último, tomando como principal argumento que en el contexto de las votaciones electrónicas en línea es determinante la eficiencia en el protocolo entre el votante y los servidores, no dejamos de lado, la propuesta de un esquema de seguridad basado totalmente en la criptografía sobre curvas elípticas.

Referencias

[1] M. Asadpour and R. Jalili. Double Voting Problem of Some Anonymous E-Voting Schemes. *Journal of Information Science and Engineering*, 25(3):895–906, 2009.

[2] L. Babai. On Lovász Lattice Reduction and the Nearest Lattice Point Problem (Shortened Version). In *Proceedings of the 2nd Symposium of Theoretical Aspects of Computer Science*, STACS '85, pages 13–20. Springer-Verlag, 1985.

[3] F. Baiardi, A. Falleni, R. Granchi, F. Martinelli, M. Petrocchi, and A. Vaccarelli. SEAS, A Secure E-Voting Protocol: Design and Implementation. *Computers and Security*, 24(8):642–652, 2005.

[4] D. Balzarotti, G. Banks, M. Cova, V. Felmetsger, R.A. Kemmerer, W.K. Robertson, F. Valeur, and G. Vigna. Are Your Votes Really Counted?: Testing the Security of Real-World Electronic Voting Systems. In *International Symposium on Software Testing and Analysis*, pages 237–248, 2008.

[5] P. Barreto, H. Y. Kim, B. Lynn, and M. Scott. Efficient Algorithms for Pairing-Based Cryptosystems. In *CRYPTO 02: Proceedings of the 22nd Annual International Cryptology Conference on Advances in Cryptology*, volume 2442, pages 354–368, London, UK, 2002. Springer-Verlag.

[6] P. Barreto, R. Lindner, and R. Misoczki. Decoding Square-free Goppa Ccodes Over $\mathbb{F}_p$. Cryptology ePrint Archive, Report 2010/372, 2010. http://eprint.iacr.org/.

[7] P. Barreto and M. Naehrig. Pairing-Friendly Elliptic Curves of Prime Order. In *Selected Areas in Cryptography*, volume 3897 of *Lecture Notes in Computer Science*, pages 319–331. Springer, 2006.

[8] M. Bellare and P. Rogaway. Random Oracles are Practical: A Paradigm for Designing Efficient Protocols. In *ACM Conference on Computer and Communications Security*, pages 62–73, 1993.

[9] J. L. Beuchat, J. E. González-Díaz, S. Mitsunari, E. Okamoto, F. Rodríguez-Henríquez, and T. Teruya. High-Speed Software Implementation of the Optimal Ate Pairing over Barreto-Naehrig Curves. In *Pairing*, volume 6487 of *Lecture Notes in Computer Science*, pages 21–39. Springer, 2010.

[10] I. Blake, G. Seroussi, N. Smart, and J. W. Cassels. *Advances in Elliptic Curve Cryptography (London Mathematical Society Lecture Note Series)*. Cambridge University Press, 2005.

[11] A. Boldyreva. Threshold Signatures, Multisignatures and Blind Signatures Based on the Gap-Diffie-Hellman-Group Signature Scheme. In *PKC '03: Proceedings of the 6th International Workshop on Theory and Practice in Public Key Cryptography*, volume 2567 of *Lecture Notes in Computer Science*, pages 31–46. Springer-Verlag, 2003.

[12] D. Boneh and M. K. Franklin. Identity-Based Encryption from the Weil Pairing. *Journal on Computing*, 32(3):586–615, 2003.

[13] D. Boneh, B. Lynn, and H. Shacham. Short Signatures from the Weil Pairing. In *ASIACRYPT01*, volume 2248, pages 514–532, 2001.

[14] J. Callas, L. Donnerhacke, H. Finney, D. Shaw, and R. Thayer. RFC-4880 OpenPGP Message Format, November 2007. (ObsoletesRFC1991, RFC2440)(Status: PROPOSED STANDARD).

[15] J. Camenisch, J. M. Piveteau, and M. Stadler. Blind Signatures Based on the Discrete Logarithm Problem. *In Advances in Crypology, EUROCRYPT '94*, 950(0):428–432, 1994.

[16] Z. Cao and L. Liu. A Strong RSA Signature Scheme and Its Application. *Software Engineering, Artificial Intelligence, Networking, and Parallel/Distributed Computing*, 01(0):111–115, 2007.

[17] S. Chatterjee, D. Hankerson, and A. Menezes. On the Efficiency and Security of Pairing-Based Protocols in the Type-1 and Type-4 Settings. In *Arithmetic of Finite Fields*, volume 6087 of *Lecture Notes in Computer Science*, pages 114–134. Springer Berlin / Heidelberg, 2010.

[18] D. Chaum. Security Without Identification: Transaction Systems to Make Big Brother Obsolete. *Communications of the ACM*, 28(10):1030–1044, 1985.

[19] D. Chaum. Blinding for Unanticipated Signatures. In *Advances in Cryptology, EUROCRYPT 87*, volume 304 of *Lecture Notes in Computer Science*, pages 227–233. Springer Berlin / Heidelberg, 1988.

[20] D. Chaum, R. L. Rivest, and A. Sherman. Crypto 82. In *Advances in Cryptology 1981 to 1997*, volume 1440 of *Lecture Notes in Computer Science*, pages 13–19. Springer Berlin / Heidelberg, 1999.

[21] Y. Choie, E. Jeong, and E. Lee. Efficient Identity-Based Authenticated Key Agreement Protocol from Pairings. *Applied Mathematics and Computation*, 162(1):179–188, 2005.

[22] J. S. Chou, Y. Chen, and J. C. Huang. A Novel Secure Electronic Voting Protocol Based on Bilinear Pairings. Cryptology ePrint Archive, Report 2006/342, 2006. http://eprint.iacr.org/.

[23] Y. F. Chung and Z. Y. Wu. Approach to Designing Bribery-Free and Coercion-Free Electronic Voting Scheme. *Journal of Systems and Software*, 82(12):2081–2090, 2009.

[24] L. F. Cranor and R. Cytron. SENSUS: A Security-Conscious Electronic Polling System for the Internet. In *Hawaii International Conference on System Sciences (3)*, pages 561–570, 1997.

[25] I. Dåmgard, J. Groth, and G. Salomonsen. The Theory and Implementation of an Electronic Voting System. In *Secure Electronic Voting*, pages 77–100. Kluwer Academic Publishers, 2003.

[26] W. Diffie and M. E. Hellman. New Directions in Cryptography. *IEEE Transactions on Information Theory*, 22(6):644–654, 1976.

[27] L. J. {Dominguez Pérez}. *Developing an Automatic Generation Tool for Cryptographic Pairing Functions*. PhD thesis, Dublin City University. Faculty of Engineering and Computing. School of Computing., 2011.

[28] T. ElGamal. A Public Key Cryptosystem and a Signature Scheme Based on Discrete Logarithms. *IEEE Transactions on Information Theory*, 31(4):469–472, 1985.

[29] G. Frey, M. Muller, and H. G. Ruck. The Tate Pairing and the Discrete Logarithm Applied to Elliptic Curve Cryptosystems. *IEEE Transactions on Information Theory*, 45(5):1717 –1719, 1999.

[30] G. Frey and H. G. Rück. A Remark Concerning m-Divisibility and the Discrete Logarithm in the Divisor Class Group of Curves. *Mathematics of Computation*, 206(62):865–874, 1994.

[31] A. Fujioka, T. Okamoto, and K. Ohta. A Practical Secret Voting Scheme for Large Scale Elections. In *Advances in Cryptology - AUSCRYPT 92, Workshop on the Theory and Application of Cryptographic Techniques*, pages 244–251, 1992.

[32] S. D. Galbraith, X. Lin, and M. Scott. Endomorphisms for Faster Elliptic Curve Cryptography on a Large Class of Curves. In *EUROCRYPT*, volume 5479 of *Lecture Notes in Computer Science*, pages 518–535. Springer, 2009.

[33] S. D. Galbraith and M. Scott. Exponentiation in Pairing-Friendly Groups Using Homomorphisms. In *Pairing*, volume 5209 of *Lecture Notes in Computer Science*, pages 211–224. Springer, 2008.

[34] R.P. Gallant, R.J. Lambert, and S.A. Vanstone. Faster Point Multiplication on Elliptic Curves with Efficient Endomorphisms. In *CRYPTO*, volume 2139 of *Lecture Notes in Computer Science*, pages 190–200. Springer, 2001.

[35] W. Gao, X. Wang, G. Wang, and F. Li. One-Round ID-Based Blind Signature Scheme without ROS Assumption. Cryptology ePrint Archive, Report 2007/007, 2007. http://eprint.iacr.org/.

[36] S. Goldwasser, S. Micali, and R. L. Rivest. A Digital Signature Scheme Secure Against Adaptive Chosen-Message Attacks. *Society for Industrial and Applied Mathematics Journal on Computing*, 17(2):281–308, 1988.

[37] D. Hankerson, A. Menezes, and S. Vanstone. *Guide to Elliptic Curve Cryptography*. Springer-Verlag New York, Inc., 2003.

[38] L. Harn. Cryptanalysis of the Blind Signatures Based on the Discrete Logarithm Problem. *Electronics Letters*, 31(14):1136, 1995.

[39] F. Hess. Efficient Identity Based Signature Schemes Based on Pairings. In *SAC '02: Revised Papers from the 9th Annual International Workshop on Selected Areas in Cryptography*, pages 310–324, London, UK, 2003. Springer-Verlag.

[40] F. Hess, N. P. Smart, and F. Vercauteren. The Eta Pairing Revisited. *IEEE Transactions on Information Theory*, 52(10):4595–4602, 2006.

[41] P. Horster, M. Michels, and H. Petersen. Cryptanalysis of the Blind Signatures Based on the Discrete Logarithm Problem. *Electronics Letters*, 31(21):1827, 1995.

[42] S. H. Hwang, H. A. Wen, and T. Hwang. On the Security Enhancement for Anonymous Secure E-Voting Over Computer Network. *Computer Standards and Interfaces*, 27(2):163–168, 2005.

[43] D. Jena, S. Kumar, and B. Majhi. A Novel Untraceable Blind Signature Based on Elliptic Curve Discrete Logarithm Problem. *International Journal of Computer Science and Network Security*, 7(6):269–275, 2007.

[44] A. Joux. A One Round Protocol for Tripartite Diffie Hellman. *Journal of Cryptology*, 17(4):263–276, 2004.

[45] M. Joye and J. Quisquater. Efficient Computation of Full Lucas Sequences. *Electronics Letters*, 32(6):537 –538, 1996.

[46] A. Juels, D. Catalano, and M. Jakobsson. Coercion-Resistant Electronic Elections. In *Proceedings of the 2005 ACM workshop on Privacy in the Electronic Society*, Workshop on Privacy in the Electronic Society '05, pages 61–70, New York, NY, USA, 2005. ACM.

[47] J. Katz and T. Lindell. *Introduction to Modern Cryptography*. Chapman & Hall/Crc Cryptography and Network Security Series, 2007.

[48] B. Kharchineh and M. Ettelaee. A New Electronic Voting Protocol Using a New Blind Signature Scheme. In *Proceedings of the 2010 Second International Conference on Future Networks*, ICFN 10, pages 190–194. IEEE Computer Society, 2010.

[49] N. Koblitz. Elliptic Curve Cryptosystems. *Mathematics of Computation*, 177(48):203–209, 1987.

[50] G. Kumar-Verma. New ID-based Fair Blind Signatures. Cryptology ePrint Archive, Report 2008/093, 2008. http://eprint.iacr.org/.

[51] M. Kutylowski and F. Zagórski. Verifiable Internet Voting Solving Secure Platform Problem. In *International Workshop on Security*, volume 4752 of *Lecture Notes in Computer Science*. Springer, 2007.

[52] C. C. Lee, W. P. Yang, and M. S. Hwang. Untraceable Bblind Signature Schemes Based on Discrete Logarithm Problem. *Fundamenta Informaticae*, 55(3-4):307–320, 2002.

[53] C. T. Li, M. S. Hwang, and Y. C. Lai. A Verifiable Electronic Voting Scheme Over the Internet. In *Proceedings of the 2009 Sixth International Conference on Information Technology: New Generations*, pages 449–454, 2009.

[54] I. Lin, M. Hwang, and C. Chang. Security Enhancement for Anonymous Secure E-Voting Over a Network. *Computer Standards and Interfaces*, 25(2):131–139, 2003.

[55] L. López-García, F. Rodríguez-Henríquez, and M. León-Chávez. An E-voting Protocol Based on Pairing Blind Signatures. In *International Conference on Security and Cryptography, Porto, Portugal*, SECRYPT 08, 2008.

[56] L. López-García, F. Rodríguez-Henríquez, and M. León-Chávez. Sistema Electrónico de Votación basado en firmas a ciegas con emparejamientos. In *X Reunión Española sobre Criptología y Seguridad de la Información*, XRECSI, Salamanca, España, 2008.

[57] L. López-García, F. Rodríguez-Henríquez, and L. Martínez-Ramos. A Comparative Performance Analysis of Nine Blind Signature Schemes. In *5th International Conference on Electrical Engineering, Computing Science and Automatic Control*, CCE 08, 2008.

[58] L. López-García, F. Rodríguez-Henríquez, and L. Martínez-Ramos. A Software Performance Comparison of Blind Signature Schemes. In *New Trends in Electrical Engineering Automatic Control, Computing and Communication Sciences*, Chapter 27, pages 459–483. Logos Verlag, Berlin, 2010.

[59] W. Mao. *Modern Cryptography: Theory and Practice*. Prentice Hall Professional Technical Reference, 2003.

[60] L. Martínez-Ramos, L. López-García, and F. Rodríguez-Henríquez. Achieving Identity-Based Cryptography in a Personal Digital Assistant Device. *Submitted to: Journal of Applied Research and Technology*, 2010.

[61] A. Menezes, T. Okamoto, and S. A. Vanstone. Reducing Elliptic Curve Logarithms to Logarithms in a Finite Field. *IEEE Transactions on Information Theory*, 39(5):1639–1646, 1993.

[62] A. Menezes, P. C. Van Oorschot, and S. A. Vanstone. *Handbook of Applied Cryptography*. CRC Press, 2001.

[63] V. S. Miller. Use of Elliptic Curves in Cryptography. In *Advances in Cryptology*, volume 218 of *Lecture Notes in Computer Science*, pages 417–426. Springer, 1985.

[64] V. S. Miller. The Weil Pairing, and its Efficient Calculation. *Journal of Cryptology*, 17(4):235–261, 2004.

[65] C. D. Mote. Report of the National Workshop on Internet Voting: Issues and Research Agenda. In *Proceedings of the 2002 Annual National Conference on Digital Government Research*, pages 1–59. Digital Government Research Center, 2002.

[66] Siguna Müller. On the Computation of Square Roots in Finite Fields. *Designs Codes and Cryptography*, 31:301–312, 2004.

[67] NIST. Digital Signature Standard (DSS). Federal Information Processing Standards Publications (FIPS PUBS), 1994. http://www.itl.nist.gov/fipspubs/fip186.htm.

[68] NIST. Secure Hash Standard (SHS). Federal Information Processing Standards Publications (FIPS PUBS), 2008. http://csrc.nist.gov/publications/fips/fips180-3/fips180-3_final.pdf.

[69] L. Nitschke. Secure Internet Voting Based on Paper Ballots. In *ICISS '08: Proceedings of the 4th International Conference on Information Systems Security*, pages 102–115, 2008.

[70] K. Nyberg and R. A. Rueppel. A New Signature Scheme Based on the DSA Giving Message Recovery. In *ACM Conference on Computer and Communications Security*, pages 58–61, 1993.

[71] K. Peng. A Hybrid E-Voting Scheme. In *ISPEC '09: Proceedings of the 5th International Conference on Information Security Practice and Experience*, pages 195–206, Berlin, Heidelberg, 2009. Springer-Verlag.

[72] K. Peng and F. Bao. A Design of Secure Preferential E-Voting. In *VOTE-ID '09: Proceedings of the 2nd International Conference on E-Voting and Identity*, pages 141–156, Berlin, Heidelberg, 2009. Springer-Verlag.

[73] D. Pointcheval. Strengthened Security for Blind Signatures. In *EUROCRYPT98*, pages 391–405, 1998.

[74] D. Pointcheval and J. Stern. Security Arguments for Digital Signatures and Blind Signatures. *Journal of Cryptology*, 13(3):361–396, 2000.

[75] G. Qadah and R. Taha. Electronic Voting Systems: Requirements Design and Implementation. *Computer Standards and Interfaces*, 29(3):376–386, 2006.

[76] D. K. Rappe. Homomorphic Cryptosystems and their Applications. Cryptology ePrint Archive, Report 2006/001, 2006. http://eprint.iacr.org/.

[77] R. L. Rivest, A. Shamir, and L. M. Adleman. A Method for Obtaining Digital Signatures and Public-Key Cryptosystems. *Communications of the ACM*, 21(2):120–126, 1978.

[78] F. Rodríguez-Henríquez, D. Ortíz, and P. García. Yet Another Improvement Over the Mu-Varadharajan E-Voting Protocol. *Computer Standards and Interfaces*, 29(4):471–480, 2007.

[79] F. Rodríguez-Henríquez, N. A. Saqip, A. Díaz-Pérez, and C. K. Koç. *Cryptographic Algorithms on Reconfigurable Hardware (Signals and Communication Technology)*. Springer-Verlag New York, Inc., 2006.

[80] R. Sakai, K. Ohgishi, and M. Kasahara. Cryptosystems Based on Pairing. In *In 2000 Symposium on Cryptography and Information Security*, pages 26–28, Okinawa, Japan, 2000.

[81] M. Scott. MIRACL – Multiprecision Integer and Rational Arithmetic C/C++ Library.

[82] A. Shamir. Identity-Based Cryptosystems and Signature Sschemes. In *Proceedings of CRYPTO 84 on Advances in cryptology*, pages 47–53, New York, NY, USA, 1985. Springer-Verlag New York, Inc.

[83] N. P. Smart and E. J. Westwood. Point Multiplication on Ordinary Elliptic Curves over Fields of Characteristic Three. *Applicable Algebra in Engineering Communication Computing*, 13(6):485–497, 2003.

[84] IEEE Standards Documents. IEEE p1363: Standard Specifications for Public Key Cryptography. Draft Version D18. IEEE, 2004. http://grouper.ieee.org/groups/1363/.

[85] D. R. Stinson. *Cryptography: Theory and Practice*. Chapman and Hall/CRC, 2006.

[86] J. Taverne, A. Faz-Hernández, D. F. Aranha, F. Rodríguez-Henríquez, D. Hankerson, and J. López. Software Implementation of Binary Elliptic Curves: Impact of the Carry-less Multiplier on Scalar Multiplication. Cryptology ePrint Archive, Report 2011/170, 2011. http://eprint.iacr.org/.

[87] S. Tuecke, V. Welch, D. Engert, L. Pearlman, and M. Thompson. Internet X.509 Public Key Infrastructure (PKI) Proxy Certificate Profile. RFC 3820 (Proposed Standard), 2004.

[88] F. Vercauteren. Optimal Pairings. Cryptology ePrint Archive, Report 2008/096, 2008. http://eprint.iacr.org/.

[89] L. Wang, J. Guo, and M. Luo. A More Effective Voting Scheme Based on Blind Signature. volume 4456 of *Lecture Notes in Computer Science*, pages 1507–1510, 2006.

[90] M. Wang, P. Wei, H. Zhang, and Y. Zheng. Optimal Pairing Revisited. Cryptology ePrint Archive, Report 2009/564, 2009. http://eprint.iacr.org/.

[91] L. C. Washington. *Elliptic Curves: Number Theory and Cryptography*. Discrete Mathematics and Its Applications. Chapman and Hall/CRC, 2003.

[92] K. Weldemariam, R. A. Kemmerer, and A. Villafiorita. Formal Specification and Analysis of an E-Voting System. In *International Conference on Availability, Reliability, and Security*, pages 164–171, 2010.

[93] L. C. Wu. Analysis of Traceability Attack on Camenisch et al.'s Blind Signature Schemes. In *Proceedings of the 2006 ACM Symposium on Information, Computer and Communications Security*, ASIACCS '06, pages 366–366. ACM, 2006.

[94] T. Wu and J. R. Wang. Comment: A New Blind Signature Based on the Discrete Logarithm Problem for Untraceability. *Applied Mathematics and Computation*, 170(2):999–1005, 2005.

[95] C. Yang, C. Lin, and H. Yang. Improved Anonymous Secure E-Voting Over a Network. *e-Government and Security of Information*, 15(2):181–195, 2004.

[96] F. Zhu. New Digital Signature Scheme Attaining Immunity to Adaptative Chosen-Message Attack. *Chinese Journal of Electronics*, 10(4):484–486, 2001.

Apéndice A

Algoritmos de firma a ciegas

A.1. Firma a ciegas de Cao y Liu

Algoritmo A.1: Genera (signatario), Cao y Liu [16]

Entrada: Parámetro de seguridad ℓ.
Salida: Llave privada (p, q), llave pública (N, X, g).

1 Elegir aleatoriamente dos números primos p y q, de $\ell/2$ bits de longitud, con $p = 2p' + 1, q = 2q' + 1$;
2 $N = pq$;
3 Seleccionar aleatoriamente dos generadores de residuos cuadráticos $X, g \in \mathbb{Z}_N$;
4 **return** (p, q, N, X, g)

Algoritmo A.2: Oculta (usuario), Cao y Liu [16]

Entrada: El mensaje m, la llave pública (N, X, g).
Salida: El mensaje oculto $(e', \bar{y})$.

1 $h = H(m)$;
2 Seleccionar aleatoriamente a, d, e de 257 bits de longitud;
3 $e' = ade$;
4 $\bar{y} \equiv (Xg^{H_1(h||e||X)})^d \bmod N$;
5 **return** $(e', \bar{y})$

Algoritmo A.3: Firma (signatario), Cao y Liu [16]

Entrada: El par $(e', \bar{y})$, la llave privada (p, q).
Salida: La firma a ciegas $(\hat{y})$.

1 Dado $\phi(N) \equiv (p-1) \cdot (q-1)$;
2 $d' \equiv (1/e') \bmod \phi(N)$;
3 $\hat{y} \equiv (\bar{y})^{d'} \bmod N$;
4 **return** $\hat{y}$

Algoritmo A.4: Descubre (usuario), Cao y Liu [16]

Entrada: La firma a ciegas $\hat{y}$, el factor de opacidad a.
Salida: El valor y.

1 $y \equiv \hat{y}^a \bmod N$;
2 **return** y

Algoritmo A.5: Verifica, Cao y Liu [16]

Entrada: La firma (y, e), el mensaje m, la llave pública (N, X, g).
Salida: {Válida/Rechazada}.

1 $h = H(m)$;
2 $X' \equiv y^e g^{-H_1(h||e||X)} \bmod N$;
3 **if** $X = X'$ **then**
4 **return** *Válida*;
5 **else**
6 **return** *Rechazada*;
7 **end**

A.2. Esquemas de firma a ciegas de Camenisch

Algoritmo A.6: Genera (signatario), Camenisch et al. [15]

Entrada: Un primo p, un primo q divisor de $(p-1)$, un generador α de orden q.
Salida: Las llaves (x, y), la llave de sesión $(k, \hat{r})$.

1 Seleccionar aleatoriamente $x \in \mathbb{Z}_q^*$;
2 Calcular $y \equiv \alpha^x \bmod p$;
3 Para cada usuario, seleccionar aleatoriamente $k \in \mathbb{Z}_q^*$;
4 Calcular $\hat{r} \equiv \alpha^k \bmod p$;
5 **return** $(x, y, k, \hat{r})$

Algoritmo A.7: Oculta (usuario), Camenish et al. propuesta 1 [15]

Entrada: Mensaje m, la llave de sesión $\hat{r}$.
Salida: El mensaje oculto $\hat{h}$, el valor r, los factores de opacidad (a, b).

1 $h = H(m)$;
2 Seleccionar aleatoriamente $a, b \in \mathbb{Z}_q^*$;
3 $r = \hat{r}^a \alpha^b \bmod p$;
4 $\hat{h} = \alpha h \hat{r} r^{-1} \bmod q$;
5 **return** $(\hat{h}, r, a, b)$

Algoritmo A.8: Firma (signatario), Camenisch et al. propuesta 1 [15]

Entrada: El mensaje oculto $\hat{h}$, la llave privada x, la llave de sesión k.
Salida: La firma a ciegas $\hat{s}$.

1 $\hat{s} = k\hat{h} + \hat{r}x \bmod q$;
2 **return** $\hat{s}$

Algoritmo A.9: Descubre (usuario), Camenisch et al. propuesta 1 [15]

Entrada: La firma oculta $\hat{s}$, la llave de sesión $\hat{r}$, el valor r, el mensaje m, el factor de opacidad b.
Salida: El valor s.

1 $h = H(m)$;
2 $s = \hat{s} r \hat{r}^{-1} + bh \bmod q$;
3 **return** s

Algoritmo A.10: Verifica, Camenisch et al. propuesta 1 [15]

Entrada: La firma (r, s), el mensaje m, la llave pública y.
Salida: {Válida/Rechazada}.

1 $h = H(m)$;
2 $r' \equiv (\alpha^s y^{-r})^{h^{-1}} \bmod q$;
3 **if** $r = r'$ **then**
4 **return** *Válida*;
5 **else**
6 **return** *Rechazada*;
7 **end**

Algoritmo A.11: Oculta (usuario), Camenisch et al. propuesta 2 [15]

Entrada: Mensaje m, la llave de sesión $\hat{r}$.
Salida: El mensaje oculto $\hat{h}$, el valor r, los factores de opacidad (a, b).

1 Seleccionar aleatoriamente $a, b \in \mathbb{Z}_q^*$;
2 $r \equiv m\alpha^a \hat{r}^b \bmod p$;
3 $\hat{m} \equiv rb^{-1} \bmod q$;
4 **return** $(\hat{m}, r, a, b)$

Algoritmo A.12: Firma (signatario), Camenisch et al. propuesta 2 [15]

Entrada: El mensaje oculto $\hat{m}$, la llave privada x, la llave de sesión k.
Salida: La firma a ciegas $\hat{s}$.

1 $\hat{s} \equiv \hat{m}x + k \bmod q$;
2 **return** $\hat{m}$

Algoritmo A.13: Descubre (usuario), Camenisch et al. propuesta 2 [15]

Entrada: La firma a ciegas $\hat{s}$, los factores de opacidad a, b.
Salida: El valor s.

1 $s \equiv \hat{s}b + a \bmod q$;
2 **return** s

Algoritmo A.14: Verifica, Camenisch et al. propuesta 2 [15]

Entrada: La firma (r, s), la llave pública y.
Salida: El mensaje m

1 $m' \equiv (\alpha^{-s} y^r) r \bmod p$;

A.3. Firma a ciegas de Jena, Kumar y Majhi sobre curvas elípticas

Algoritmo A.15: Genera (signatario), Jena et al. [43]

Entrada: Una curva elíptica E, punto generador P de orden n.
Salida: Las llaves (d, Q), la llave de sesión $(k, \hat{R})$.

1 Seleccionar aleatoriamente $d \in \mathbb{Z}_n^*$;
2 $Q = dP$;
3 Para cada usuario, seleccionar aleatoriamente $k \in \mathbb{Z}_n^*$;
4 $\hat{R} = kP$;
5 **return** $(d, Q, k, \hat{R})$

Algoritmo A.16: Oculta (usuario), Jena et al. [43]

Entrada: Mensaje m, la llave de sesión $\hat{R}$, una función picadillo H.
Salida: El mensaje oculto $\hat{h}$, los factores de opacidad (a, b) y el punto R.

1 Seleccionar aleatoriamente $a, b \in \mathbb{Z}_q^*$;
2 $h = H(m)$;
3 $R = a\hat{R} + bP = (x_r, y_r)$;
4 $r \equiv x_r \bmod n$;
5 $\hat{h} \equiv ah\hat{r}r^{-1} \bmod n$;
6 **return** $(\hat{h}, R, a, b)$

Algoritmo A.17: Firma (signatario), Jena et al. [43]

Entrada: El mensaje oculto $\hat{h}$, la llave privada d, la llave de sesión k.
Salida: La firma a ciegas $\hat{s}$.

1 $\hat{s} \equiv d\hat{r} + k\hat{h} \bmod n$;
2 **return** $\hat{s}$

Algoritmo A.18: Descubre (usuario), Jena et al. [43]

Entrada: La firma oculta $\hat{s}$, la llave de sesión $\hat{r}$, el valor r, el mensaje m, el factor de opacidad b.
Salida: El valor s.

1 $h = H(m)$;
2 $s \equiv \hat{s}\hat{r}^{-1}r + bh \bmod n$;
3 **return** s

Algoritmo A.19: Verifica, Jena et al. [43]

Entrada: La firma (r, s), el mensaje m, la llave pública Q.
Salida: {Válida/Rechazada}.

```
1 R es el punto con coordenada r;
2 if P = s^{-1}(rQ + H(m)R) then
3     return Válida;
4 else
5     return Rechazada;
6 end
```

A.4. Firma a ciegas de Gao et al.

Algoritmo A.20: Oculta (usuario), Gao [35]

Entrada: El mensaje m.
Salida: El mensaje oculto P'_m y el factor de opacidad r_1.

```
1 Seleccionar aleatoriamente r_1 ∈ Z_r;
2 P'_m = r_1 H_1(m);
3 return (P'_m, r_1)
```

Algoritmo A.21: Firma (signatario), Gao [35]

Entrada: El mensaje oculto P'_m, la llave privada S_{ID}.
Salida: La firma a ciegas (A', B', C').

```
1 Seleccionar aleatoriamente x_ID ∈ Z_r;
2 A' = x_ID P'_m;
3 B' = x_ID^{-1} S_ID;
4 C' = x_ID P;
5 return (A', B', C')
```

Algoritmo A.22: Descubre, Gao et al. [35]

Entrada: La firma a ciegas (A', B', C'), las llaves públicas P_{pub} y Q_{ID}, el valor r_1.
Salida: {Válida/Rechazada}.

if $e(A', P) = e(P'_m, C') \,\&\, e(Q_{ID}, P_{pub}) = e(B', C')$ **then**
    Seleccionar aleatoriamente $r_2 \in \mathbb{Z}_r$;
    $A = r_2 r_1^{-1} A'$;
    $B = r_2^{-1} B'$;
    $C = r_2 C'$;
    return (A, B, C);
else
    return *Rechazada*;
end

Algoritmo A.23: Verifica, Gao [35]

Entrada: La firma (S', c'), el mensaje m, las llaves públicas P_{pub} y Q_{ID}.
Salida: {Válida/Rechazada}.

if $e(A, P) = e(H_2(m), C) \,\&\, e(Q_{ID}, P_{pub}) = e(B, C)$ **then**
    return *Válida*;
else
    return *Rechazada*;
end