Centro de Investigación y de Estudios Avanzados
del Instituto Politécnico Nacional

UNIDAD ZACATENCO

DEPARTAMENTO DE INGENIERÍA ELÉCTRICA

SECCIÓN DE COMPUTACIÓN

Diseño y Desarrollo de un Sistema para Elecciones Electrónicas Seguras (SELES)

Tesis que presenta:

Claudia Patricia García Zamora

Para obtener el grado de:

Maestría en Ciencias

En la especialidad de:

Ingeniería Eléctrica

Opción de Computación

Director de tesis

Francisco José Rambó Rodríguez Henríquez

México, D.F. Septiembre 2005

Índice general

Resumen

Los avances recientes en sistemas computacionales, redes y técnicas criptográficas han hecho posible considerar las elecciones en línea como una alternativa factible a las elecciones tradicionales. Hasta nuestros días, se han propuesto muchos protocolos para elecciones electrónicas, desafortunadamente sólo algunos de ellos se han implementado.

Por tal motivo, en esta tesis presentamos el diseño y la implementación del Sistema para ELecciones Electrónicas Seguras (SELES); el cual fue desarrollado para ser funcional a mediana escala y cuyo protocolo de votación constituye una mejora al propuesto por Lin-Hwang-Chang [18].

El objetivo de SELES es ser un sistema para votaciones electrónicas en línea que posea las propiedades deseables de los sistemas convencionales, como lo son: exactitud, democracia, privacidad, verificación, simplicidad o convenciencia, flexibilidad y detección de dos o más votos emitidos por un mismo elector. Además de ofrecer características que ningún esquema tradicional tiene, como lo son la seguridad y la privacidad física.

Adicionalmente, SELES ha sido diseñado para ser robusto en relación con las fallas que pueden presentarse en las comunicaciones, dándole así cierto grado de tolerancia a fallos.

Abstract

Recent advances in communication networks and cryptographic techniques have made possible on-line voting as a feasible alternative to conventional elections. Until today several protocols for electronic voting have been proposed, unfortunately only a few of them have been implemented.

In this thesis is presented the design and the implementation of a secure electronic voting system for medium scale on-line elections (SELES, *Sistema para ELecciones Electrónicas Seguras*); improving the scheme proposed by Lin-Hwang-Chang [18].

The goal of SELES is to be an electronic voting system that accomplishes the desired properties of conventional voting systems, like accuracy, democracy, privacy, verifiability, simplicity, flexibility, and double voting detection. SELES also offers two important characteristics: the physical security and physical privacy, two services that are not in a traditional voting system.

Additionally, SELES has been designed to deal with communication failures, giving it a certain degree of robustness.

Índice de figuras

Índice de cuadros

Capítulo 1

Introducción

Algunos gobiernos y organizaciones democráticas necesitan contar con mecanismos para que sus miembros puedan ejercer su derecho a votar. Tradicionalmente, las elecciones sirven como un mecanismo oficial para que los ciudadanos expresen sus preferencias sobre los gobernantes o dirigentes que desea tener, mientras que las encuestas se utilizan para conocer la opinión pública. En ambas, la privacidad y la seguridad son usualmente deseables, sin embargo no siempre se alcanzan estas características a un precio razonable, ya que los mecanismos que aseguran la seguridad y la privacidad de una elección pueden ser muy costosos para los administradores e inconvenientes para los votantes.

Además, una porción significativa de la ciudadanía con derecho a votar en elecciones de cualquier tipo (gubernamentales y no gubernamentales), frecuentemente no lo hacen. Una de las principales razones para no votar es el hecho de que las personas encuentran inconveniente tener que desplazarse a un lugar determinado para ejercer ese derecho, aun cuando estos lugares suelen designarse cerca de su residencia o de su lugar de trabajo. Tampoco se debe olvidar que a veces las personas no pueden presentarse a votar porque se encuentran fuera de la ciudad a causa del trabajo o porque están de vacaciones. O bien, porque si se encuentran en la ciudad, sus actividades cotidianas les impiden ir a las casillas correspondientes, en el caso de elecciones gubernamentales.

Actualmente, con el crecimiento rápido de las redes computacionales y los avances en técnicas criptográficas, el voto en línea ofrece una alternativa razonable a las elecciones convencionales.

El voto electrónico en línea podría permitir a los votantes participar en una elección desde cualquier lugar que cuente con acceso a red (intranet ó Internet) por medios alámbricos o inalámbricos. De esta manera, las personas serían capaces de emitir sus votos desde el trabajo, la escuela o desde la comodidad de sus hogares, según sea el caso. Aunado a lo anterior, se tendría el valor agregado que las personas puedan emitir un voto contando con la privacidad física, lo cual quiere decir, que éstas podrían participar de manera activa en las elecciones sin necesidad de ser vistas por los demás votantes o personal administrativo.

Resulta importante señalar que esta última característica es imposible de obtener en las elecciones tradicionales.

Sin embargo, si no se cuenta con las medidas de seguridad pertinentes, estos sistemas pueden ser comprometidos causando resultados electorales fraudulentos o violaciones a la privacidad de los participantes.

Por lo tanto, considerando: que 1) las elecciones son eventos fundamentales en organizaciones democráticas, 2) que actualmente se cuenta con grandes avances en técnicas criptográficas, y 3) que en los últimos años las redes computacionales han tenido un gran desarrollo tecnológico, resulta claro ver que las elecciones electrónicas son una alternativa viable a las elecciones tradicionales [17]. Además de que las primeras pueden ofrecer características deseables que las segundas son incapaces de proporcionar por su misma naturaleza, como la privacidad y seguridad física.

Es por ello, que en este trabajo de tesis se ha diseñado e implementado un Sistema para Elecciones Electrónicas Seguras (SELES) utilizando diversas técnicas criptográficas y dispositivos inalámbricos.

El sistema se desarrolló utilizando criptografía de llave pública, certificados digitales, firmas a ciegas con RSA, esquemas de firma digital, estampas de tiempo y funciones *hash*. Estos conceptos serán explicados más adelante.

SELES está diseñado para poder ser ejecutado en dispositivos móviles, como computadoras portátiles con tarjeta de red inalámbrica ó asistentes personales digitales (PDA's *Personal Digital Asistants*).

Otro aspecto importante que conviene resaltar es el hecho que SELES fue desarrollado considerando su funcionalidad correcta para un padrón mediano, es decir, aproximadamente cinco mil participantes. Esto fue debido a que para un número mayor de electores, los protocolos propuestos requieren un mayor número de servidores para recabar, procesar y contar los votos; lo cual reduce la eficiencia del sistema.

Por último, cabe mencionar que el sistema ha sido implementado para poder usarse en dispositivos inalámbricos. Lo cual es un valor agregado que puede ser aprovechado por personas que poseen este tipo de herramientas, para que de esta manera puedan emitir su voto desde un lugar que cuente con un punto de acceso a Internet: como una escuela, oficina, biblioteca, o centro comercial. Sin olvidar que la movilidad se está volviendo un requisito primordial en nuestros días y cada vez más personas, ajenas a la computación, hacen uso de éstos.

Este documento está organizado de la siguiente manera: en el Capítulo 2 se describen los principales conceptos utilizados en cuanto a criptografía asimétrica o de llave pública,

incluyendo algunas herramientas asimétricas y los esquemas de firma digital. En el Capítulo 3 se presentan algunos aspectos importantes del servicio de autenticación, tales como: los ataques al mismo, la descripción de los certificados digitales, y la infraestructura de llave pública (PKI). A continuación, en el Capítulo 4 se presenta información relacionada con las *elecciones electrónicas*; como son conceptos básicos y algunos de los esquemas que se han propuesto. Además de que se incluye una descripción detallada del esquema de Lin–Hwang–Chang y del protocolo que se implementó para SELES. Después, se detalla el diseño y la implementación del sistema en el Capítulo 5; abarcando las tres diferentes autoridades y la entidad *elector*. Las pruebas realizadas, los resultados obtenidos y el análisis y evalución final se exponen en el Capítulo 6. Por último, en el Capítulo 7 se discuten las conclusiones a las que se llegó al finalizar el desarrollo de este trabajo.

Capítulo 2

Criptografía asimétrica

Los inicios de la criptografía se remontan a miles de años atrás; sin embargo, en la década de los años 60's el gran auge de las computadoras y de los sistemas de comunicación provocó un aumento en la investigación enfocada a la seguridad de la información, y por ende a la criptografía.

Una forma muy simple de definir la *criptografía* es decir que es la ciencia de ocultar el contenido de los mensajes de forma segura [31]. Aunque una definición más formal nos dice que es el estudio de técnicas matemáticas relacionadas con aspectos de seguridad de la información, tales como autenticación de entidades y de datos, confidencialidad, e integridad de datos [23].

Por lo tanto, la criptografía provee un conjunto de técnicas para codificar mensajes de forma tal que dichos mensajes puedan ser almacenados y transmitidos en forma segura. Por ejemplo, la criptografía puede ser utilizada para almacenar información confidencial (que un intruso no pueda leerla), o para transmitir mensajes por canales inseguros o poco confiables en forma totalmente segura. Además de mantener la confidencialidad, la criptografía puede ser utilizada para asegurar la integridad de los datos a almacenar o transmitir, es decir que éstos no puedan ser modificados y que estos cambios pasen desapercibidos. También se puede verificar la autenticidad de un mensaje y, usando firmas digitales, se puede lograr que un mensaje no sea repudiado, es decir, que la persona que lo envió no pueda negar su origen. En resumen, los principales servicios ofrecidos por la criptografía son:

1. Confidencialidad
2. Integridad de datos
3. Autenticación de entidades y de datos
4. No repudio

Existen varias maneras de clasificar los sistemas de cifrado. Una de ellas los clasifica en simétricos y asimétricos.

Los sistemas simétricos son aquellos que utilizan la misma llave para cifrar y descrifrar un documento. El principal servicio que ofrecen es la confidencialidad.

Los sistemas de cifrado asimétricos, llamados también *de llave pública*, hacen uso de dos llaves diferentes. Una es la llave pública, la cual puede darse a conocer a cualquier persona; y la otra, llamada llave privada, debe ser mantenida en secreto. Para enviar un mensaje, el remitente usa la llave pública del destinatario para cifrar el mensaje. Una vez que el mensaje ha sido cifrado, únicamente con la llave privada del destinatario se podrá descifrar, ni siquiera la entidad que originalmente cifró el mensaje puede volver a descifrarlo. Por ello, se puede dar a conocer sin problema la llave pública para que todo aquel que se quiera comunicar confidencialmente con el destinatario lo pueda hacer. Este tipo de sistemas ofrecen de manera óptima los servicios de integridad de datos, autenticación de entidades y de datos, y el no repudio.

En nuestros días existen varios algoritmos de llave pública importantes, por ejemplo: RSA, DSA, ElGamal y criptografía de curvas elípticas (CCE). Estos algoritmos basan su seguridad en problemas matemáticos computacionalmente difíciles de resolver, tales como:

- *RSA:* En la dificultad matemática que representa el problema de factorizar números enteros grandes (de más de 512 bits).
- *DSA y ElGamal:* En el problema del logaritmo discreto.
- *CCE:* En el alto grado de dificultad que supone resolver, al igual que los dos anteriores, el problema del logaritmo discreto, pero en el grupo Abeliano formado por curvas elípticas definidas sobre campos finitos [7].

En este capítulo se presentan algunos algoritmos y técnicas relevantes de la criptografía asimétrica o de llave pública. En la sección 2.1 se muestran algunas de las herramientas criptográficas utilizadas como bloques básicos en la criptografía asimétrica. La sección 2.2 contiene una descripción de los diferentes esquemas de firma digital que existen actualmente. Y por último en la sección 2.3 se exponen las características de las firmas a ciegas.

2.1. Herramientas criptográficas

En esta sección se presentamos algunas herramientas o técnicas que son usadas por la criptografía para alcanzar sus objetivos. Cada una de ellas tiene una aplicación y un propósito muy específico dentro del ámbito criptográfico.

2.1.1. Preliminares matemáticos

Antes de revisar las técnicas criptográficas incluídas en esta sección, se explicarán algunas definiciones importantes para su correcta comprensión: [23]

- *Números enteros*: El conjunto de números enteros $\{..., -3, -2, -1, 0, 1, 2, 3, ...\}$ se denota por el símbolo Z.
- *Divisibilidad*: Sean a y b dos enteros. Entonces se dice que a *divide* a b (en otras palabras: a es un *divisor* de b, o a es un *factor* de b), si existe un entero c tal que $b = ac$. Si a divide a b, se escribe: $a|b$.
- *Máximo Común Divisor*: Un entero positivo d es el máximo común divisor de los enteros a y b, denotado $d = \text{MCD}(a, b)$, si d es el entero positivo mayor que divide a ambos a y b.
- *Primos relativos*: Dos enteros a y b se dice que son *primos relativos* si $\text{MCD}(a, b) = 1$.
- *Congruencia*: Sea n un entero positivo. Si a y b son enteros, entonces se dice que a es *congruente* con b módulo n, escrito como $a \equiv b (\text{mód } n)$, si n divide $(a - b)$. El entero n es llamado el *módulo* de la congruencia.
- *Propiedades de las congruencias*: Para todos los a, a_1, b, b_1, $c \in Z$, las siguientes propiedades se cumplen:
  1. $a \equiv b$ (mód n) si y sólo si a y b tienen el mismo residuo cuando son divididos por n.
  2. (*reflexibilidad*) $a \equiv a$ (mód n).
  3. *(simetría)* Si $a \equiv b$ (mód n) entonces $b \equiv a$ (mód n).
  4. *(transitividad)* Si $a \equiv b$ (mód n) y $b \equiv c$ (mód n), entonces $a \equiv c$ (mód n).
  5. Si $a \equiv a_1$ (mód n) y $b \equiv b_1$ (mód n), entonces $a + b \equiv a_1 + b_1$ (mód n) y $ab \equiv a_1 b_1$ (mód n).
- *Teorema chino del residuo*: Si los enteros $n_1, n_2, ..., n_k$ son primos relativos entre sí, entonces el sistema de congruencias simultáneas

$$\begin{aligned} x &\equiv a_1 (\text{mód} n_1) \\ x &\equiv a_2 (\text{mód} n_2) \\ &\vdots \\ x &\equiv a_k (\text{mód} n_k) \end{aligned}$$

  tiene solución única módulo $n = n_1 n_2 \cdots n_k$. Este teorema se explicará con mayor detalle en la siguiente subsección.

2.1.2. Teorema chino del residuo

En muchas situaciones, es útil romper una congruencia módulo n en un sistema de congruencias módulo factores de n. Considérese el siguiente ejemplo. Supongamos que se conoce un número x el cual satisfase $x = 25$ mód 42. Lo anterior significa que se puede escribir $x = 25 + 42k$ para algún entero k. Reescribiendo 42 como $7 \cdot 6$, se obtiene $x = 25 + 7(6k)$, lo cual implica que $x = 25 = 4$ mód 7. De la misma manera se tiene $x = 25 + 6(7k)$, y entonces $x = 25 = 1$ mód 6. Por lo tanto,

$$x = 25 \text{ mód } 42 \Rightarrow \{ \begin{array}{l} x = 4 \text{ mód } 7 \\ x = 1 \text{ mód } 6 \end{array}$$

El *TRC* muestra que este procedimiento es reversible; en otras palabras, un sistema de congruencias puede ser reemplazado por una sola bajo ciertas condiciones.

Teorema chino del residuo [33]: Para el caso de un sistema de dos congruencias, considere que el máximo común divisor entre m y n es 1, es decir, $\text{MCD}(m, n) = 1$. Y dados los valores a y b, existe exactamente una solución x mód mn para las siguientes congruencias simultáneas:

$$\begin{array}{c} x = a \text{ mód } m, \\ x = b \text{ mód } n. \end{array}$$

Demostración: La existencia de dos enteros s y t tales que $ms + nt = 1$. Entonces $ms = 1$ mód n, y $nt = 1$ mód m. Sea $x = bms + ant$. De esta manera $x = ant = a$ mód m, y $x = bms = b$ mód n. Supóngase que x_1 es otra solución. Por lo tanto, $x = x_1$ mód m, y $x = x_1$ mód n, así $x - x_1$ es múltiplo de m y n.

Para demostrar la unicidad, se hace uso del siguiente lema:

Lema: Sean m y n enteros con $\text{MCD}(m, n) = 1$. Si un entero c es múltiplo de ambos m y n; entonces c es múltiplo de mn.

Demostración: Supongamos que $c = mk = nl$. Y dado que previamente quedó establecido que $\text{MCD}(m, n) = 1$, es posible localizar los enteros s y t, tales que $ms + nt = 1$. Al multiplicar esta última expresión por c obtendremos: $c = cms + cnt = mnls + mnkt = mn(ls + kt)$, de donde se puede ver claramente que c es un múltiplo de mn.

Unicidad: Para finalizar la demostración del teorema consideraremos $c = x - x_1$ en el lema, para así encontrar que $x - x_1$ es múltiplo de mn. Por lo tanto, $x = x_1$ mód mn. Esto significa que cualquiera de las dos soluciones x del sistema de congruencias son congruentes módulo mn, como se había indicado.

2.1.3. Funciones *hash*

En el esquema de firma digital, los mensajes que se intercambian pueden tener un gran tamaño, hecho que dificulta el proceso de cifrado. Por ello, en la realidad no se cifra el mensaje entero sino un resumen del mismo, obtenido aplicando al mensaje una función *hash*.

Partiendo de un mensaje determinado que puede tener cualquier tamaño, éste se convierte mediante la función *hash* en un mensaje con una dimensión fija (generalmente de 128 ó 160 bits). Para ello, el documento original se divide en varias partes, cada una de las cuales tendrá el mismo tamaño, y una vez dividido se combinan elementos tomados de cada una de las partes resultantes de la división para formar el compendio o *hash*, que también tendrá una dimensión fija y constante. Este compendio del mensaje de dimensión fija es el que se cifrará utilizando la llave privada del emisor del documento.

Las propiedades matemáticas que son deseables para una función *hash* son las siguientes: [33]

- La función $h()$ debe ser fácil de calcular para cualquier m.
- La función *hash* debe ser de un sólo sentido o dirección (*sólo ida*), es decir, si se conoce $h(m)$ encontrar m debe implicar calcular todos los m posibles.
- La función *hash* debe ser resistente a las colisiones [23], es decir, no debe ser posible (computacionalmente) encontrar m y m' con $m \neq m'$ tales que $h(m) = h(m')$.

Las funciones *hash* son usadas principalmente para resolver problemas relacionados con la integridad de los mensajes, así como en los procesos de verificación de la autenticidad de mensajes y de su origen.

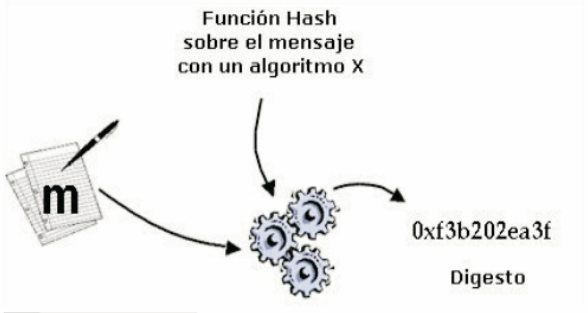

Figura 2.1: Función *hash*

También son utilizadas en criptosistemas de llave pública para cifrar la llave privada mediante una contraseña elegida por el dueño del par de llaves. Esto es, se le pide al

usuario que indique una contraseña (alrededor de 10 caracteres), y después se le aplica una función *hash*; el compendio resultante funciona como llave simétrica para cifrar la llave privada a través de un algoritmo simétrico (DES, TripleDES, etc). Y así, el usuario puede cifrar y descifrar su llave privada con sólo recordar la contraseña inicial que proporcionó (ver figura 2.2).

Entre los algoritmos más importantes de las funciones *hash* están: MD5 y SHA-1. El Algoritmo MD5 es el resultado de una serie de mejoras sobre el algoritmo MD4, diseñado por Ron Rivest, el cual procesa los mensajes de entrada en bloques de 512 bits, y produce una salida de 128 bits. El algoritmo SHA-1, roto recientemente, fue desarrollado por la NSA, para ser incluido en el estándar DSS (Digital Signature Standard)[33]. Produce tramas de 160 bits, a partir de bloques de 512 bits del mensaje original.

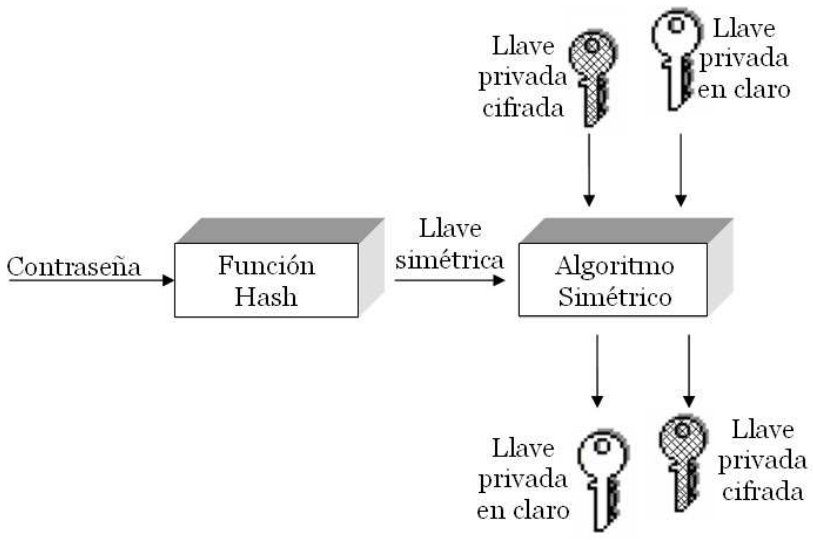

Figura 2.2: Cifrado / descifrado de una llave privada

2.1.4. Algoritmo RSA

El criptosistema RSA, llamado así en honor a sus inventores Ronald Rivest, Adi Shamir y Leonard Adleman, es el algoritmo asimétrico más utilizado en la actualidad. Como se mencionó anteriormente, basa su seguridad en el problema de la factorización de números enteros grandes.

Para generar el par de llaves se deben realizar los siguientes pasos: [33]

1. Generar dos números primos grandes de manera aleatoria, p y q.

2. Calcular $n = pq$, y $\phi(n) = (p-1)(q-1)$.

3. Elegir un entero aleatorio e, $1 < e < \phi(n)$, que sea primo relativo de $\phi(n)$, es decir, que el máximo común divisor de e y $\phi(n)$ sea 1.

4. Generar un número d, $1 < d < \phi(n)$, tal que $ed \equiv 1$ mód $\phi(n)$.

5. Finalmente, la llave pública será (e, n); mientras que la llave privada será (d, n).

Ahora bien, para cifrar un mensaje m, éste debe estar dentro del intervalo $[0, n-1]$. Si m cumple lo anterior, calculando: $c = m^e$ mód n, se obtiene el mensaje cifrado c. Y para recuperar el mensaje original basta con realizar la siguiente ecuación: $m = c^d$ mód n.

2.2. Esquemas de firma digital

Técnicamente, la firma digital es un conjunto o bloque de caracteres que viaja junto a un documento, archivo o mensaje y que puede acreditar quién es el autor o emisor del mismo (lo que se denomina autenticación), y que nadie ha manipulado o modificado el mensaje en el transcurso de la comunicación (integridad).

La firma digital equivale funcionalmente a la firma autógrafa en cuanto a la identificación del autor del que procede el mensaje.

La aparición y desarrollo de las redes de computadoras, de las que Internet es el ejemplo más notorio, ha supuesto la posibilidad de intercambiar entre personas geográficamente distantes todo tipo de mensajes, incluidos los de contenido privado. Estos mensajes plantean el problema de acreditar tanto la autenticidad como la autoría de los mismos.

Concretamente, para que dos individuos (un votante y una autoridad electoral, o un empresario y un consumidor) puedan intercambiarse entre sí mensajes electrónicos de carácter privado que sean mínimamente fiables y puedan, en consecuencia, dar a las partes involucradas la confianza y la seguridad que necesita el tráfico de dicho tipo de mensajes, se deben cumplir los siguientes requisitos:

1. *Autenticación*, que implica poder atribuir de forma indudable el mensaje electrónico recibido a una determinada persona como autora del mensaje.

2. *Integridad*, que implica la certeza de que el mensaje recibido por B (receptor) es exactamente el mismo mensaje emitido por A (emisor), sin que haya sufrido alteración alguna durante el proceso de transmisión de A hacia B.

3. *No repudio o no rechazo en origen*, que implica que el emisor del mensaje (A) no pueda negar en ningún caso que el mensaje ha sido enviado por él.

Pues bien, la firma digital es un procedimiento técnico que basándose en técnicas criptográficas trata de dar respuesta a esa triple necesidad apuntada anteriormente, a fin de

posibilitar el tráfico comercial electrónico.

La firma digital se basa en la utilización combinada de dos técnicas distintas, que son la criptografía asimétrica o de llave pública para cifrar los mensajes y el uso de las llamadas funciones *hash* o funciones resumen (ver figura 2.3).

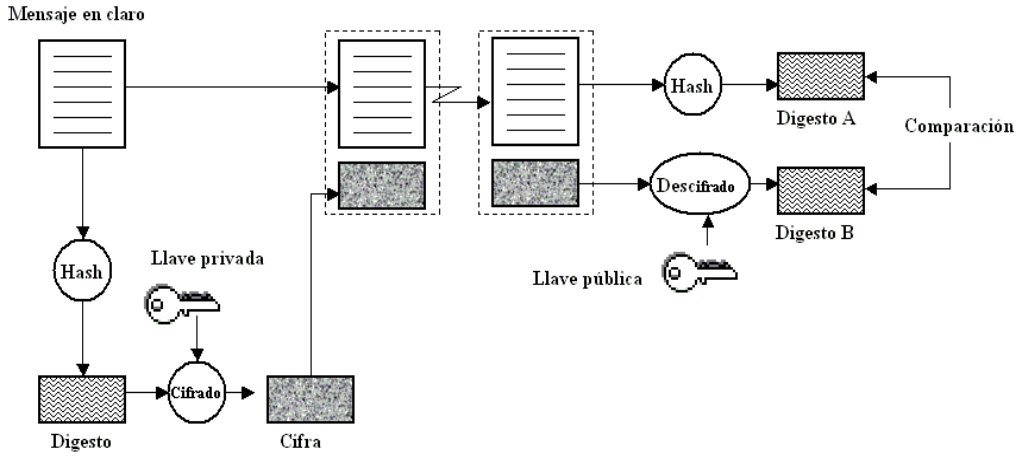

Figura 2.3: Esquema de firma digital

En ocasiones, además de garantizar la procedencia de los mensajes electrónicos que se intercambian por medio de internet y la autenticidad o integridad de los mismos, puede ser conveniente garantizar también su confidencialidad, que no es un requisito esencial de la firma digital sino accesorio de la misma. Ello implica que el mensaje no pueda ser leído por terceras personas distintas de A (emisor) y de B (receptor) durante su proceso de transmisión, es decir, que se debe tener la certeza de que el mensaje enviado por A únicamente será leído por B y no por personas ajenas a la relación que mantienen A y B.

En tales casos, también se acude al cifrado del mensaje con el par de llaves, pero de manera diferente al mecanismo propio y característico de la firma digital. Para garantizar la confidencialidad del mensaje, el cuerpo del mismo (no el *hash* o resumen) se cifra utilizando la llave pública de B (receptor), quien al recibir el mensaje lo descifrará utilizando para ello su llave privada (la llave privada de B). De esta manera se garantiza que únicamente B pueda descifrar el cuerpo del mensaje y conocer su contenido.

En el resto de esta sección se examinarán los esquemas de firma digital relacionados con el desarrollo de este proyecto de tesis.

2.2.1. Esquema de firma ElGamal

El esquema de firma ElGamal genera firmas digitales con apéndice sobre mensajes binarios de tamaño arbitrario, y requiere una función *hash* ($h()$) para obtener los compendios

de los mensajes a firmar.

El proceso de generación de llaves para A es el siguiente:

1. Obtener un número primo largo p y un generador α.
2. Elegir un número entero aleatorio a, tal que $1 \leq a \leq p-2$.
3. Calcular $y = \alpha^a$ mód p.
4. Una vez hecho lo anterior, la llave pública de A es (p, α, y), mientras que a es su llave privada.

Para que A firme un mensaje m debe hacer lo siguiente:

1. Elegir aleatoriamente un entero secreto k tal que $1 \leq k \leq p-2$. Y con MCD$(k, p-1)$=1.
2. Calcular $r = \alpha^k$ mód p, y obtener $h(m)$.
3. Calcular $s = k^{-1}\{h(m) - ar\}$ mód $(p-1)$.
4. La firma de A para el mensaje m es (r, s).

Para verificar la firma (r, s), B tiene que seguir el procedimiento que se muestra a continuación:

1. Obtener la llave pública de A, es decir, (p, α, y).
2. Verificar que se cumpla $1 \leq r \leq p-1$; si no es así, la firma se rechaza.
3. Calcular $v_1 = y^r r^s$ mód p.
4. Calcular $h(m)$, y $v_2 = \alpha^{h(m)}$ mód p.
5. La firma es declarada válida si y sólo si $v_1 = v_2$. [23]

2.2.2. Esquema de firma DSA

En agosto de 1991, el *Instituto Nacional de Estándares y Tecnología de los Estados Unidos* (NIST) propuso el *Algoritmo de Firma Digital* denominado DSA, por sus siglas en inglés (*Digital Signature Algorithm*). Tres años más tarde en 1994, DSA se convirtió en un estándar llamado *Estándar de Firma Digital* (DSS), y además adquirió el reconocimiento de cualquier gobierno como esquema de firma digital [23].

Este algoritmo es una variante de esquema ElGamal, es un mecanismo de firma digital con apéndice, y requiere de una función *hash* para firmar los compendios de los mensajes.

El proceso de generación de un par de llaves DSA, para una entidad A, es como se muestra a continuación [33]:

1. Elegir un número primo q, tal que $2^{159} < q < 2^{160}$.
2. Escoger un número t que cumpla $0 \leq t \leq 8$, y elegir otro número primo p, tal que $2^{511+64t} < p < 2^{512+64t}$; con la propiedad de que q divida a $(p-1)$.
3. Sea g una raíz primitiva módp, y sea $\alpha \equiv g^{(p-1)/q}$ mód p. Entonces $\alpha^q \equiv 1$ mód p.
4. Elegir de manera aleatoria un número entero a, tal que $1 \leq a \leq q-1$. Y calcular $y = \alpha^a$ mód p.
5. Finalmente, la llave pública de A es (p, q, α, y); y la privada es a.

Para que A firme un mensaje m debe seguir el siguiente proceso:

1. Elegir de manera secreta un número entero k, tal que $0 < k < q$, y aplicarle una función *hash* al mensaje, $h(m)$.
2. Calcular $r = (\alpha^k \text{ mód } p) \text{ mód } q$.
3. Calcular $s = k^{-1}\{h(m) + ar\} \text{ mód } q$.
4. La firma de A para m es (r, s). Y ésta es enviada a B junto con m.

Ahora bien, para verificar la validez de la firma (r, s), B debe:

1. Conocer la llave pública de A, es decir (p, q, α, y). Y calcular el compendio del mensaje, $h(m)$.
2. Verificar que las desigualdades $0 < r < q$ y $0 < s < q$ se cumplan; si no es así, la firma es rechazada.
3. Calcular $w = s^{-1} \text{ mód } q$.
4. Calcular $u_1 = w \cdot h(m) \text{ mód } q$, y $u_2 = r \cdot w \text{ mód } q$.
5. Calcular $v = (\alpha^{u_1} y^{u_2} \text{ mód } p) \text{ mód } q$.
6. La firma se acepta si y sólo si $v = r$.

2.2.3. Firmas con RSA

Debido a que los procesos de cifrado/descifrado en RSA (explicados anteriormente en la sección 2.1.4) son una función biyectiva, las firmas digitales pueden ser creadas invirtiendo el orden de dichos procedimientos.

Por lo tanto, para que A genere una firma del mensaje m con RSA debe realizar los siguiente pasos:

1. Generar un par de llaves. Como se mencionó previamente, A debe generar dos números primos grandes p y q. Calcular $n_A = pq$ y $\phi(n_A) = (p-1)(q-1)$. Elegir e_A, $1 < e_A < \phi(n_A)$, tal que $\text{MCD}(e_A, \phi(n_A)) = 1$. Y calcular d_A, tal que $e_A d_A \equiv 1 \text{ mód } \phi(n_A)$. Así, su llave pública es (e_A, n_A), y su llave privada es (d_A, n_A).

2. La firma se obtiene calculando primero el *hash* del mensaje $h(m)$, y después $y \equiv h(m)^{d_A} \text{ mód } n_A$.

3. Y los valores (m, y) se envían al receptor de la firma B.

Para que B realice la verificación de la firma (m, y) debe:

1. Conocer (e_A, n_A), es decir, la llave pública de A.

2. Aplicarle la función *hash* al mensaje para obtener $h(m)$.

3. Calcular $z \equiv y^{e_A} \text{ mód } n_A$. Si $z = h(m)$, entonces B acepta la firma como válida; de otra manera la rechaza por ser inválida. [23]

2.3. Firmas a ciegas

Las *firmas a ciegas* son un tipo especial de firmas digitales en las que se firma algo que no se conoce. Para hacer firmas a ciegas se utilizan factores de opacidad, para ocultar el mensaje original que se requiere que esté firmado, y así la autoridad no pueda conocer lo que está firmando.

Por lo tanto, el propósito de una firma a ciegas es evitar que el firmante B conozca el mensaje que firma; y así posteriormente, sea incapaz de asociar el mensaje que firmó con el remitente A.

Entonces, las firmas a ciegas tienen aplicación en varias situaciones. A continuación se mencionan dos de ellas:

- Cuando se utiliza dinero electrónico. En este caso, m representa un valor monetario que A (el cliente) tiene derecho a gastar. Y así, cuando m y $s(m)$ se presentan a B (el

banco) para efectuar el pago, B es incapaz de identificar al cliente que originalmente le dio ese dinero electrónico a firmar, pues le fue enviado de manera oculta. Lo anterior permite que la identidad de A permanezca anónima, y sus movimientos financieros no puedan ser monitoreados.

- En las elecciones electrónicas también pueden utilizarse las firmas a ciegas, ya que se requiere que B (una autoridad electoral) no conozca la identidad de A (el votante) debido a que el voto debe efectuarse de manera anónima. Sin embargo, es necesario que A demuestre que su voto m es válido. Lo cual se logra cuando A presenta ante B la firma $s(m)$. Y se sabe de antemano que B no puede asociar $s(m)$ a A, debido a que el votante previamente le envío a B su voto m pero de forma oculta para que se lo firmara.

2.3.1. Firmas a ciegas en RSA

Un esquema de firmas a ciegas es un protocolo que involucra un remitente A y un firmante B. La idea básica en un esquema basado en RSA es la siguiente: A le envía cierta información z a B, donde z está compuesto por el mensaje que se desea que firme B y por un factor de ocultamiento cifrado con la llave pública de B, es decir, $z = (m * b^e)$ mód n. B firma dicha información $s(z)$ y se la regresa a A. De la firma $s(z)$, A puede obtener la firma de B para el mensaje m, quitando el factor de ocultamiento b a $s(z)$. Pues:

$$s(z) = (m * b^e)^d \text{ mód } n = (m^d * b^{ed}) \text{ mód } n = (m^d \text{ mód } n) * b$$

Ahora bien, al dividir $s(z)$ entre b, obtendremos $s(m)$:

$$s(m) = s(z)/b = ((m^d \text{ mód } n) * b)/b = m^d \text{ mód } n$$

Al finalizar el protocolo, B no conoce el mensaje m ni la firma asociada a él $s(m)$ que ahora posee A. (Ver figura 2.4).

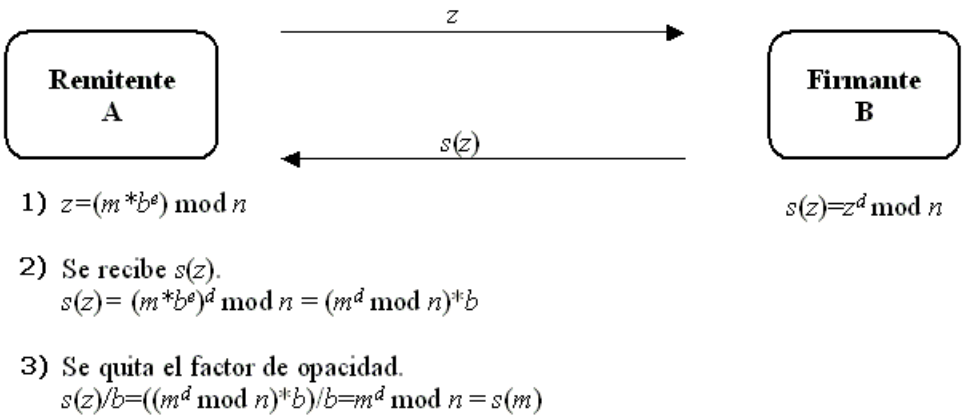

Figura 2.4: Esquema de firma a ciegas utilizando RSA

Capítulo 3

Autenticación

Como se mencionó en el capítulo anterior, los principales servicios de la Criptografía son: autenticación, confidencialidad, integridad y no repudio. Ahora bien, la autenticación es el proceso mediante el cual se verifica y asegura la identidad de las partes involucradas en una transacción. Si este proceso no se lleva a cabo existe la posibilidad de que una entidad desconocida asuma una identidad falsa, comprometiendo de esta manera la privacidad y la integridad de la información. Por lo tanto, la autenticación es fundamental en los sistemas de llave pública.

De esta manera, es claro ver que el servicio de autenticación es de suma importancia para un sistema de elecciones electrónicas. Puesto que es necesario que ambas partes, esto es, autoridades electorales y votantes, estén completamente seguros de que el intercambio de información se efectuará entre las entidades autorizadas. Por tal motivo, este capítulo se discuten los ataques a la autenticación y las técnicas que se han desarrollado para prevernirlos y evitarlos.

En la sección 3.1 se presentan algunos ataques a la autenticación. También se explican los certificados digitales y su importancia en la sección 3.2. A continuación, en la sección 3.3 se describe el concepto de autoridad certificadora y sus características. Finalmente en la sección 3.4 se incluyen la definición y propiedades de la infraestructura de llave pública (PKI).

3.1. Ataques

En la sección 2.2 se explicó que con el uso de firmas digitales se logra proveer el servicio de autenticación. Sin embargo, se han desarrollado diversos ataques al esquema de firma digital. Por lo cual, ha sido necesario hacer uso de diversas herramientas y técnicas para evitar o eliminar los efectos que provocan dichos ataques.

Una posible clasificación a los ataques por autenticación es la siguiente:

1. **Pasivos**: Estos ataques consisten en escuchar el tráfico que circula por una red, con la intención de obtener cierta información de una manera no autorizada.

2. **Activos**: Son aquéllos que no sólo interceptan la información sino que también manipulan los datos que circulan por la red (ver figura 3.1). Algunos ataques de este tipo son: [20]

   - *Interrupción*: Se presenta cuando se destruye una pieza de software o hardware. Es un ataque a la disponibilidad y la solución es tener un respaldo o una ruta alterna para la transmisión de la información.
   - *Intercepción*: Es un ataque directo a la confidencialidad. Se anula utilizando esquemas de cifrado.
   - *Modificación*: Ataque directo a la integridad. Se evita haciendo uso de *funciones hash*.
   - *Fabricación*: Alguien ajeno a la red se hace pasar como miembro y viola el servicio de autenticación. Lógicamente es un ataque directo a la autenticidad.

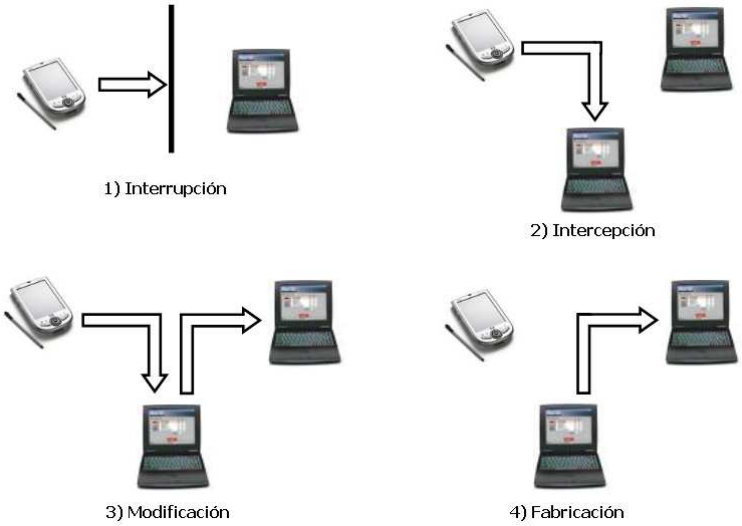

Figura 3.1: Ataques activos

Existe otro ataque, mostrado en la figura 3.2, que puede burlar el esquema de llave pública sin necesidad de romperlo, y es conocido como " intruso de enmedio " (*man in the middle*). Este ataque consiste en que si A desea realizar una transacción con B, ambas deben intercambiarse sus respectivas llaves públicas para poder cifrar la información que se va a transmitir. Sin embargo, existe una entidad C que intercepta las llaves públicas tanto de A como de B, y así C puede enviarles su propia llave pública a ambas entidades haciéndoles creer que tienen la llave pública que esperaban (es decir, A cree que tiene

la de B y viceversa). De esta manera, cuando A y B empiecen a mandar información cifrada a través de la red sólo C será capaz de descifrarla, ya que el proceso de cifrado se habrá hecho con su llave pública. Por lo tanto, es importante autenticar la identidad de la entidad con la cual se establece contacto y, a su vez, certificar de alguna manera que la llave pública sí pertenece a dicha entidad. Este tipo de ataque puede evitarse con el uso de *certificados digitales*.

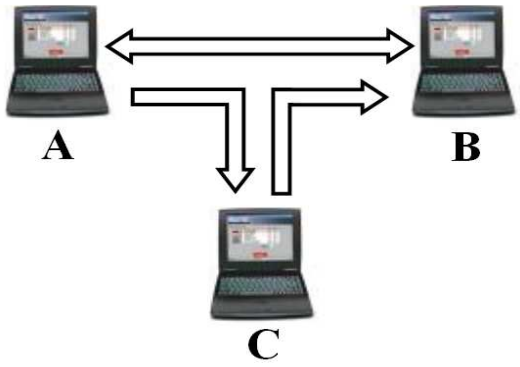

Figura 3.2: Ataque del *intruso de enmedio*

Otro ataque a la autenticación es el llamado de " replay " (*repetición*), el cual consiste en reenviar paquetes de información. Como puede verse, para realizar lo anterior no se necesita conocer la llave que se utilizó para cifrar el mensaje. Este tipo de ataque puede ocasionar graves problemas, puesto que podría darse el caso de que el mensaje que se esté reenviando sea un pago al banco, el cual se efectuará dependiendo del número de veces que haya sido reenviado dicho mensaje. Además de que se estará consumiendo valioso tiempo de procesamiento por parte del procesador, resolviendo un problema inútil. Para evitar este tipo de ataque se utilizan *contadores* o *estampas de tiempo*.

Por último se mencionará el ataque denominado " usurpación de la identidad " (*identity misbinding*). Este ataque consiste en que una entidad C puede obtener la llave pública de A y decir que le pertenece. Con esto, C tiene la capacidad de realizar transacciones haciéndose pasar por A. Además, en el caso de que se llegara a necesitar alguna firma de C para efectuar, por ejemplo, algún movimiento bancario, C podría hacerse pasar ahora como una autoridad (como un banco) y así solicitarle a A una firma para algún trámite inofensivo y tal vez imaginario. Al igual que el ataque de "intruso de enmedio", este ataque puede ser evitado con *certificados digitales* [20,26].

3.2. Certificados digitales

Un certificado de llave pública es un enlace que asocia la llave pública de una entidad y uno o más atributos referidos a su identidad. El certificado garantiza que la llave pública pertenece a la entidad identificada y que la entidad posee la correspondiente llave privada. Los certificados de llave pública se denominan comúnmente *Certificado Digital*, *ID Digital* o simplemente *Certificado*. La entidad identificada se denomina *sujeto del certificado* o *subscriptor* (por ejemplo: una persona o una empresa).

Los certificados digitales sólo son útiles si existe alguna autoridad certificadora (AC) que los valide, ya que si uno se certifica a sí mismo no hay ninguna garantía de que su identidad sea la anunciada, y por lo tanto, no debe ser aceptada por un tercero que no lo conozca.

Es importante ser capaz de verificar que una autoridad certificadora haya emitido un certificado, y detectar si un certificado es realmente válido. Para evitar la falsificación de certificados, la entidad certificadora después de autenticar la identidad de un sujeto, firma el certificado digitalmente.

Para su correcto funcionamiento, los certificados contienen además la siguiente información:

- Un identificador del propietario del certificado, que consta de su nombre, sus apellidos, su dirección de correo electrónico, datos de su empresa como el nombre de la organización, departamento, localidad, provincia y país, etc.
- Otro identificador de quién asegura su validez, que será una autoridad certificadora.
- Dos fechas, una de inicio y otra de fin del período de validez del certificado. Es decir, cuándo un certificado empieza a ser válido y cuándo deja de serlo, fecha a partir de la cual la llave pública que se incluye en él, no debe utilizarse para cifrar o firmar.
- Un identificador del certificado o número de serie, que será único para cada certificado emitido por una misma autoridad certificadora. Esto es, identificará inequívocamente a un certificado frente a todos los certificados de esa AC.
- Firma de la autoridad certificadora de todos los campos del certificado que asegura la autenticidad del mismo.

Los certificados digitales proporcionan un mecanismo criptográfico para implementar la autentificación; también proporcionan un mecanismo seguro y escalable para distribuir llaves públicas en comunidades grandes.

3.2.1. Certificados X.509

El formato de certificados X.509 es un estándar del ITU-T (*International Telecommunication Union - Telecommunication Standarization Sector*) y el ISO/IEC (*International Standards Organization / International Electrotechnical Commission*) que se publicó por primera vez en 1988. El formato de la versión 1 fue extendido en 1993 para incluir dos nuevos campos que permiten soportar el control de acceso a directorios.

Después de emplear el X.509 v.2 para intentar desarrollar un estándar de correo electrónico seguro, el formato fue revisado para permitir la extensión con campos adicionales, dando lugar al X.509 v.3, publicado en 1996 y cuyos campos se muestran en la figura 3.3.

Los elementos del formato de un certificado X.509 v.3 son:

- *Versión.* El campo de versión contiene el número de versión del certificado codificado. Los valores aceptables son 1, 2 y 3.
- *Número de serie del certificado.* Este campo es un entero asignado por la autoridad certificadora. Cada certificado emitido por una AC debe tener un número de serie único.
- *Identificador del algoritmo de firmado.* Este campo identifica el algoritmo empleado para firmar el certificado (por ejemplo RSA o DSA).
- *Nombre del emisor.* Este campo identifica la AC que ha firmado y emitido el certificado.
- *Período de validez.* Este campo indica el período de tiempo durante el cual el certificado es válido y la AC está obligada a mantener información sobre el estado del mismo. El campo consiste en una fecha inicial en la que el certificado empieza a ser válido, y una fecha después de la cual el certificado deja de serlo.
- *Nombre del sujeto.* Este campo identifica la identidad cuya llave pública está certificada en el campo siguiente. El nombre debe ser único para cada entidad certificada por una determinada AC, aunque puede emitir más de un certificado con el mismo nombre si es para la misma entidad.
- *Información de llave pública del sujeto.* Este campo contiene la llave pública, sus parámetros y el identificador del algoritmo con el que se emplea la llave.
- *Identificador único del emisor.* Este es un campo opcional que permite reutilizar nombres de emisor.
- *Identificador único del sujeto.* Este es un campo opcional que permite reutilizar nombres de sujeto.

- *Extensiones*. Este campo es opcional y sólo aparece en los certificados X.509 versión 3. Si el campo está presente, entonces el certificado contiene una o más extensiones.

Las extensiones del X.509 v.3 proporcionan una manera de asociar información adicional a sujetos, llaves públicas, etc. Un campo de extensión tiene tres partes:

1. *Identificador de extensión*. Es un identificador de objeto que proporciona la semántica y el tipo de información (cadena de texto, fecha u otra estructura de datos) para un valor de extensión.

2. *Valor de la extensión*. Este subcampo contiene propiamente el valor de la extensión.

3. *Indicador de importancia*. Es una bandera que indica si la extensión es *crítica* o *no-crítica*.

Cualquier organización puede definir una extensión privada para poder cumplir sus requerimientos específicos. Esta flexibilidad crea un nuevo inconveniente: un certificado digital X.509 versión 3 no puede ser completamente leíble por las diferentes implementaciones que soportan certificados X.509 v3. Pues, cuando alguna extensión de certificado no sea conocida por la aplicación que lo recibe, la incompatibilidad se hará presente. Por esta razón existe la bandera que indica si una extensión es *crítica* o *no-crítica*. Si la extensión es marcada como *no-crítica*, la aplicación ignora esa extensión; por otro lado, si es marcada como *crítica*, el resultado es que el certificado no puede ser utilizado debido a que se desconoce la funcionalidad de la extensión.

El ITU y el ISO/IEC han desarrollado y publicado un conjunto de extensiones estándar en un apéndice al X.509 v.3:

- *Limitaciones básicas*. Este campo indica si el sujeto del certificado es una AC y el máximo nivel de profundidad de un camino de certificación a través de esa AC.

- *Política de certificación*. Este campo contiene las condiciones bajo las que la AC emitió el certificado y el propósito del certificado.

- *Uso de la llave*. Este campo restringe el propósito de la llave pública certificada, indicando, por ejemplo, que la llave sólo se debe usar para firmar, para el cifrado de llaves, para el cifrado de datos, etc. Este campo suele marcarse como importante, ya que la llave sólo está certificada para un propósito y usarla para otro no estaría validado en el certificado.

El formato de certificados X.509 se especifica en un sistema de notación denominado *sintaxis abstracta uno* (*Abstract Sintax One* o ASN-1). Para la transmisión de los datos se aplica el DER (*Distinguished Encoding Rules* o *reglas de codificación distinguible*), que transforma el certificado en formato ASN-1 en una secuencia de octetos apropiada para la transmisión en redes reales.

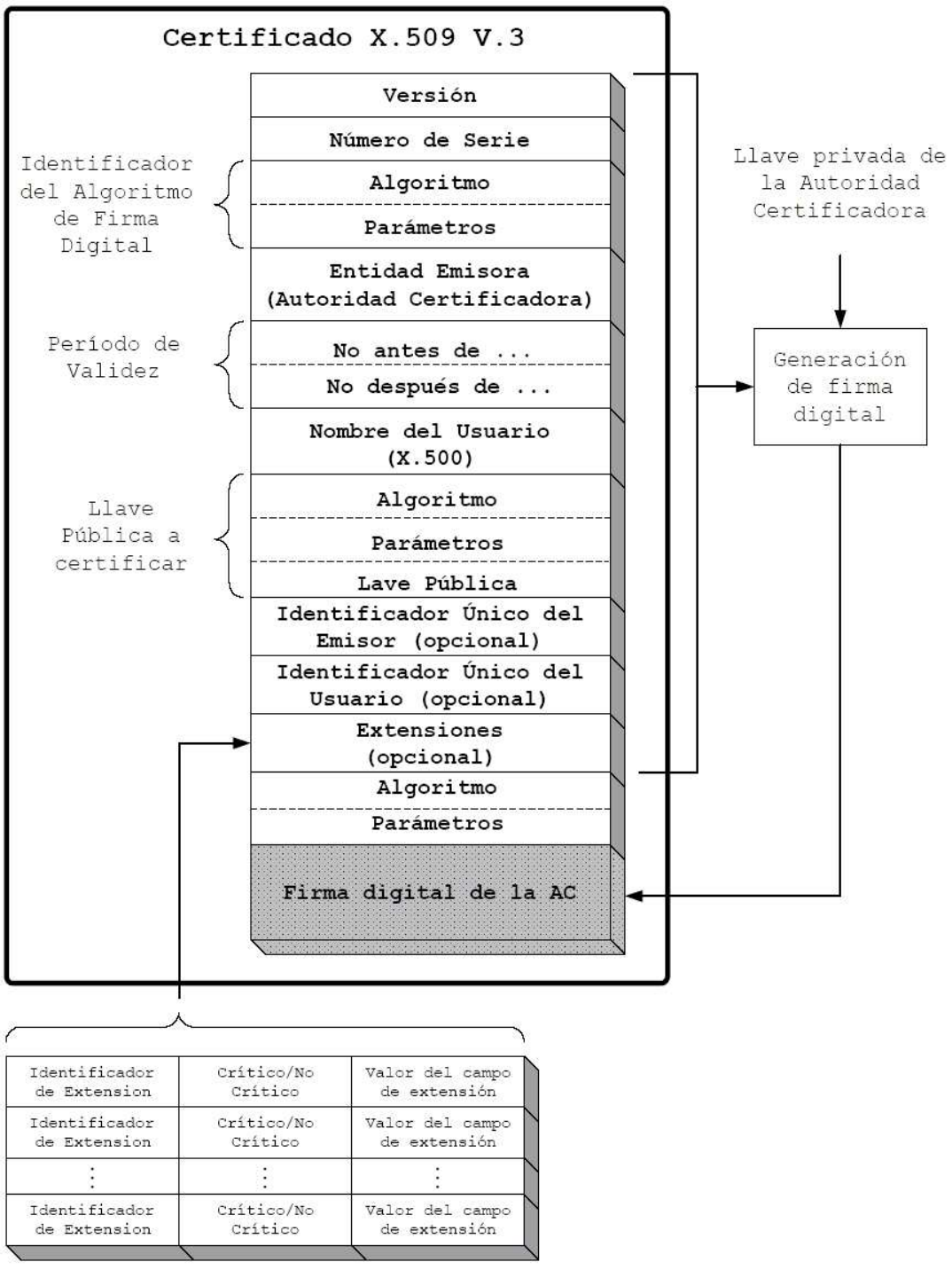

Figura 3.3: Certificado digital X.509 v.3

3.3. Autoridad certificadora

La autoridad certificadora (AC), es la entidad que firma digitalmente los certificados, asegurando su integridad y certificando la relación existente entre la llave pública con-

tenida y la identidad del propietario. En otras palabras, es la encargada de confirmar que el dueño de un certificado es realmente la persona que dice ser. La firma de la AC es la que garantiza la validez de los certificados.

Una AC puede definir las políticas especificando cuáles campos del *nombre distintivo* son opcionales y cuáles requeridos. También puede especificar requerimientos en el contenido de los campos de los certificados digitales.

En un esquema jerárquico pueden existir varias autoridades certificadoras. De esta manera, una autoridad certificadora certifica o verifica la identidad de otra AC y así sucesivamente; sin embargo, habrá un punto en que una autoridad no tendrá quién la certifique, en este caso, el certificado es firmado por uno mismo (*self-signed*). Por lo tanto, esta última AC es verificada por ella misma.

Las autoridades certificadoras deben ser entes fiables y ampliamente reconocidos que firman las llaves públicas de las personas, certificando con su propia firma la identidad del usuario. Por lo tanto, si se desea establecer una AC, se deben tomar extremadas precauciones para evitar que sus llaves privadas caigan en manos de intrusos, ya que si esto llegará a suceder se comprometería todo el sistema. Para ello se tendrán que utilizar llaves grandes y dispositivos especiales para su almacenamiento. Además, cuando se emite un certificado, se debe estar seguro de que se hace a la persona adecuada. No se puede olvidar que la autoridad certificadora es la responsable, en última instancia, de todo el proceso. Además, posee una serie de responsabilidades legales y basa operabilidad en el nivel de confianza que inspire a sus potenciales clientes [22].

Por lo tanto, la confianza de los usuarios a la AC es fundamental para el buen funcionamiento del servicio. Y el entorno de seguridad (control de acceso, cifrado, etc.) de la AC debe ser muy fuerte, en particular en lo que respecta a la protección de sus llaves privadas que utiliza para firmar los certificados que emite, como se mencionaba anteriormente.

Los usuarios pueden fácilmente identificar los certificados expedidos por la AC por medio de la comparación del nombre. Sin embargo, para asegurar que el certificado sea genuino, ellos pueden verificar la firma utilizando la llave pública de la AC, la cual debe estar disponible.

Una AC también debe expedir y procesar Listas de Revocación de Certificados (CRLs), las cuales son listas de los certificados que han sido invalidados. Los certificados pueden ser revocados por distintas razones, por ejemplo:

- Si el propietario del certificado pierde su llave privada.
- Si la compañía que posee el certificado cambia de nombre.

- Si el propietario de la llave privada abandona la empresa para la cual trabaja.

Las CRLs también deben de documentar el estado de revocación de los certificados y especificar la fecha exacta en la cual éstos fueron revocados.

Las labores básicas de una autoridad certificadora son:

- *Admisión de solicitudes*: Un usuario llena un formulario y lo envía a la AC solicitando un certificado. La generación de las llaves pública y privada son responsabilidad del usuario o de un sistema asociado a la AC.
- *Autenticación del sujeto*: Antes de firmar la información proporcionada por el sujeto, la AC debe verificar su identidad. Dependiendo del nivel de seguridad deseado y el tipo de certificado se deberán tomar las medidas oportunas para su validación.
- *Generación de certificados*: Después de recibir una solicitud y validar los datos la AC genera el certificado digital correspondiente y lo firma con su llave privada. Posteriormente lo manda al subscriptor y, opcionalmente, lo envía a un almacén de certificados para su distribución.
- *Distribución de certificados*: La entidad certificadora puede proporcionar un servicio de distribución de certificados para que las aplicaciones tengan acceso y puedan obtener los certificados de sus correspondientes subscriptores. Los métodos de distribución pueden ser: correo electrónico, servicios de directorio como el X.500 o el LDAP, etc.
- *Revocación y renovación de certificados*: La AC debe ser capaz de revocar certificados si se presentan situaciones en las que se vuelva necesario invalidarlos. Además, debe tener la facultad de renovar los certificados de los clientes que se lo soliciten.
- *Almacenes de datos*: Hoy en día existe una noción formal de *almacén* donde se guardan los certificados y la información de las revocaciones. La designación oficial de una base de datos como *almacén* tiene por objeto señalar que el trabajo con los certificados es fiable y de confianza.

Por último, es importante mencionar que la AC no sólo inserta su nombre y su firma en cada certificado que expide, sino también en la CRL que genera y mantiene actualizada constantemente.

3.4. Infraestructura de llave pública

Los algoritmos asimétricos tales como RSA, Diffie-Hellman y DSA, han revolucionado la Criptografía; sin embargo estos criptosistemas poseen ciertas carencias inherentes. Por

tal motivo, las aplicaciones prácticas que incluyen algoritmos de llave pública presentan algunos inconvenientes que sólo pueden ser evitados construyendo una infraestructura adecuada. A este tipo de infraestructura se le conoce como PKI (*Public Key Infraestructure*). Actualmente, la creación de infraestructuras de llave pública es uno de los principales objetivos de la Criptografía, especialmente por su relación con Internet y dispositivos móviles.

Cuando se implementan criptosistemas de llave pública sin el uso de infraestructuras adicionales surgen diversos problemas, los cuales pueden dividirse en cuatro áreas principales: [30]

1. *Autenticidad de la llave*: Surgen incovenientes relacionados con la autenticidad de las llaves cuando no existe algo que indique a quien pertenecen realmente. En otras palabras, cuando se utilizan los criptosistemas de llave pública sin herramientas adicionales, existe la posibilidad de que se presente el ataque *del intruso de enmedio* y el de *usurpación de la identidad*, explicados en la sección 3.1.

2. *Revocación de llaves*: Los problemas surgen cuando una entidad C roba la llave privada de A. De esta manera C es capaz de leer todos los mensajes cifrados con la llave pública de A, además de poder generar firmas a nombre de A. En el momento en que A se de cuenta de esta situación, puede protegerse generando un nuevo par de llaves y dejando de utilizar el par comprometido. Sin embargo, ¿cómo sabrán todas las entidades que mantienen una comunicación con A, a través de criptografía asimétrica, que se ha revocado el primer par de llaves?, o bien, ¿cómo conocerán la fecha exacta en que ocurrió dicha revocación?.

3. *No repudio*: El propósito de una firma digital es asegurar el no-repudio. Por lo cual, si una entidad A mantiene su llave privada en secreto, significa que nadie más puede generar una firma digital con la llave privada de A. Sin embargo, puede darse el caso de que A no acepte alguna de sus firmas digitales explicando simplemente que la llave con la cual se generó dicha firma no es la suya. El problema es que no hay forma de probar que la llave que generó la firma digital realmente pertenece a A.

4. *Aplicación de políticas*: Consideremos una situación en la que una compañía desea que cada uno de sus empleados posea un par de llaves para poder cifrar y firmar documentos. Para lograr esto se requiere que:

   - Cada empleado sea propietario de únicamente un par de llaves.
   - Todas las llaves públicas estén registradas de manera centralizada.
   - Cada empleado utilice una llave de un tamaño adecuado.
   - Cada par de llaves sea cambiado después de un período determinado de tiempo.
   - Si un empleado deja la compañía por alguna razón, su llave pública sea revocada automáticamente.

El problema del esquema anterior es que las políticas anteriormente mencionadas no pueden ser cumplidas sin el uso de herramientas y técnicas adicionales a los algoritmos asimétricos.

La Infraestructura de Llave Pública es una combinación de software, tecnologías de cifrado, y servicios que permiten proteger la seguridad de las transacciones de información sobre un sistema distribuido. PKI involucra certificados digitales, criptografía de llave pública y autoridades certificadoras dentro de una arquitectura de seguridad.

Una PKI se conforma de diversos componentes fundamentales (véase figura 3.4), tales como:

1. *Entidad final*: Es el usuario final o cualquier entidad que pueda ser identificada (personas, servidores, estaciones de trabajo, compañías, dispositivos móviles, etc.) mediante un certificado digital expedido por una autoridad certificadora.

2. *Autoridad certificadora (AC)*: Es la entidad encargada de generar los certificados digitales, y de mantener actualizada la lista de revocación (CRL). Adicionalmente puede realizar algunas funciones administrativas, aunque generalmente estas son delegadas a una o varias autoridades de registro.

3. *Autoridad de registro (AR)*: Una AR es componente opcional que puede asumir funciones administrativas de la AC. Las funciones de la AR están frecuentemente asociadas con la afiliación de las entidades finales. Se compone de una serie de elementos tecnológicos (hardware y software específico) y medios humanos (operadores de registro). En resumen, es el punto de comunicación entre los usuarios de la PKI y la autoridad certificadora.

4. *Repositorio*: Es cualquier método para el almacenamiento de certificados y listas de revocación (CRLs) generados por la autoridad certificadora. Además, permite el acceso por parte de las entidades finales a dichos elementos.

5. *Emisor CRL*: Es un componente opcional que está encargado de efectuar las tareas de actualización y publicación de las listas de revocación.

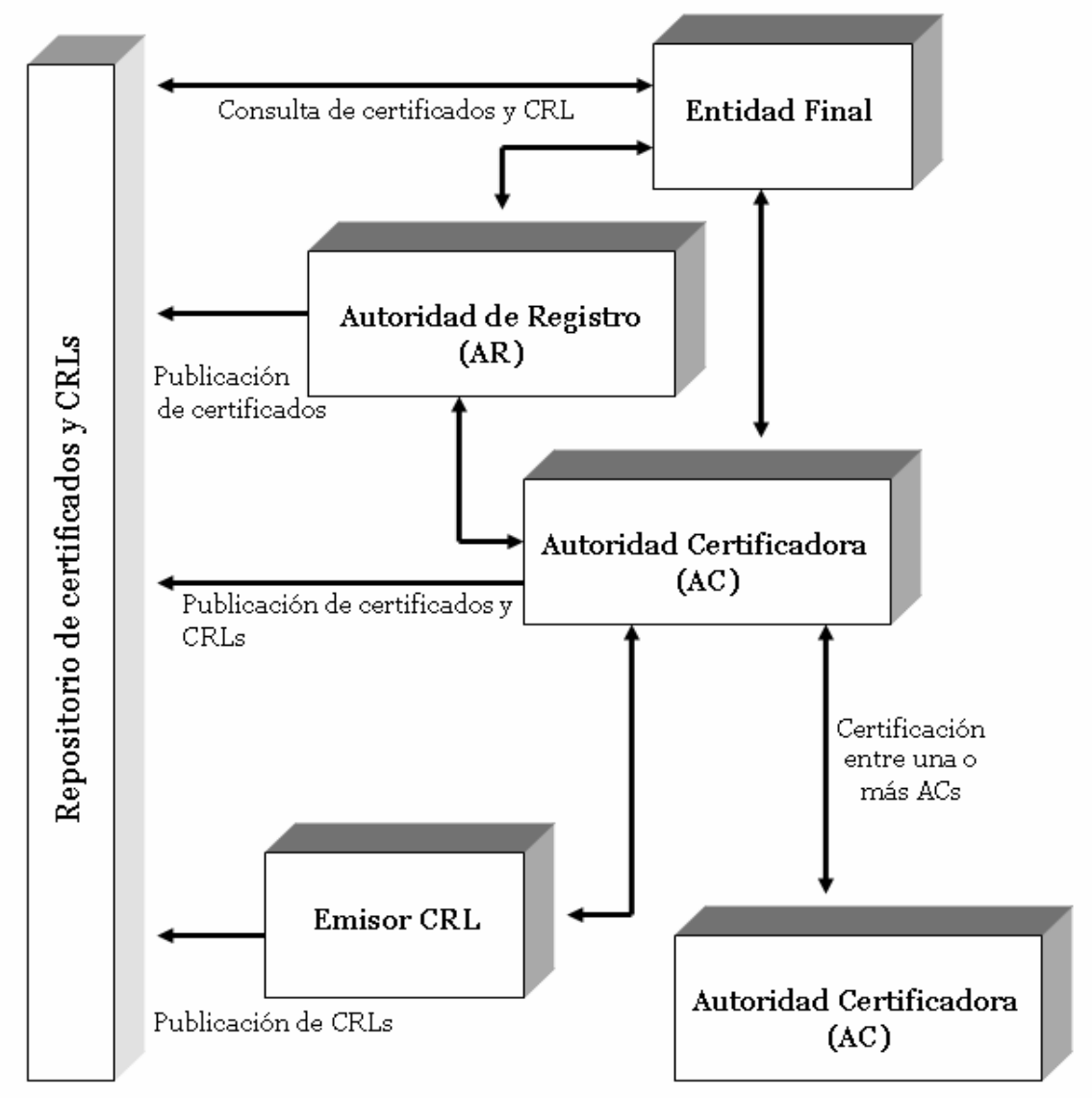

Figura 3.4: Componentes de una PKI

Capítulo 4

Elecciones electrónicas

En la sección 4.1 de este capítulo se presentan algunos conceptos básicos referentes a las elecciones electrónicas. Después, en la sección 4.2 se muestra una panorama general de los esquemas que han sido propuestos para lograr emitir votos a través de la red. En la sección 4.3 se presenta a detalle el esquema de Lin–Hwang–Chang [18], incluyendo las restricciones que éste tiene. Por último, en la sección 4.4 se expone detalladamente el protocolo implementado en nuestro proyecto de tesis.

4.1. Conceptos básicos

La elección en un régimen democrático es el procedimiento que se sigue para designar a las personas que ocuparán ciertos cargos, desempeñarán ciertas funciones o tendrán alguna posición, dentro del Estado o alguna organización a través del voto de sus conciudadanos o miembros. Por ejemplo, dentro de un estado o país se pueden elegir presidentes, senadores, diputados, gobernadores, entre otros. Y dentro de alguna organización en particular, se puede votar para escoger a los empleados del mes, los mejores accionistas, los directores académicos, los jefe de grupo, etcétera. En otros contextos, también pueden llevarse a cabo elecciones dentro de ciertos concursos, como en el denominado *el rival más débil*.

Las elecciones se caracterizan por el hecho de que el individuo elector goza de la oportunidad y de la libertad de elegir entre candidatos que representan distintas opciones. Sólo si hay dos o más candidatos entre los cuales se pueda elegir, y siempre que exista un clima de libertad para decidirse por cualquiera de ellos, se puede hablar de una elección democrática. Sin olvidar que también es necesario y fundamental contar con un sistema de votaciones justo, confiable e incorruptible. Si no existen estas condiciones, no es posible contar con elecciones libres y competitivas.

Sin embargo, no todos los miembros de la organización tienen derechos electorales. Los tienen solamente aquellos que reúnen determinados privilegios, los cuales tienden a garantizar cierto grado de capacidad reflexiva para desempeñar la función pública de elegir. De esto se sigue que el grupo de electores, llamado cuerpo electoral, no coincide cuantitati-

vamente con la población total de cierta organización; luego entonces, la responsabilidad de elegir recae no sobre el todo social, sino sobre una parte de él: sobre aquélla que, por sus condiciones particulares, está calificada para desempeñar la función reflexiva en que consiste el voto. Así por ejemplo, para elegir al *rival más débil* sólo participan las personas que en ese momento estén jugando.

Algunos tipos de elecciones pueden ser los que a continuación se mencionan:

- *Elecciones directas y elecciones indirectas:* En la votación directa, o de primer grado, el elector elige por sí mismo al candidato; en la votación indirecta o de segundo grado, vota por una lista de electores quienes a su vez eligen en definitiva a los candidatos.

- *Elecciones universales y elecciones restringidas:* Según la amplitud con la que se concede el derecho al voto, se distinguen elecciones universales y restringidas. Las primeras confieren derechos electorales a un amplio sector de la población o membresía de alguna organización en específico, con la exigencia del menor número posible de condiciones para el ejercicio del sufragio. Las segundas son aquéllas en las cuales el derecho electoral se limita a ciertos grupos de personas, ya sea que éstas estén nominalmente enumeradas, o bien que tal exclusividad derive de ciertas condiciones establecidas.

- *Elecciones obligatorias y voluntarias:* En las elecciones obligatorias la emisión del voto es para cada miembro un derecho y un deber cívico ineludible, en oposición a la elección voluntaria o facultativa, donde la emisión del voto es un derecho renunciable de cada persona. Por ejemplo en algunos países, como Brasil, las elecciones para presidente de la nación son obligatorias; mientras que en otros, como México, el no participar en las elecciones presidenciales es un derecho que los ciudadanos pueden o no ejercer.

Otra clasificación para las elecciones pueden ser de acuerdo a su alcance. De esta manera, podrían considerarse elecciones a gran escala o masivas (para miles o cientos de miles de participantes), a mediana escala (del orden de miles de participantes), y a pequeña escala (menor a 1000 participantes aproximadamente).

Elecciones electrónicas

Las elecciones electrónicas son aquéllas que requieren de medios electrónicos para llevarse a cabo. Estos medios pueden incluir computadoras, tarjetas inteligentes, redes computacionales, etc. Son elecciones en donde la intervención directa de personas disminuye significativamente, siendo reemplazada por dispositivos electrónicos.

Cuando se diseñan sistemas para elecciones electrónicas, es fundamental considerar qué características deben poseer, sin sacrificar la privacidad del votante o introducir oportunidades para cometer fraude. También se pueden considerar aquellas propiedades que son deseables, pero que no siempre son cubiertas por los sistemas tradicionales.
Por lo tanto, para este proyecto de tesis, se proponen las siguientes propiedades para el sistema a desarrollar:

1. *Privacidad del voto:* Un sistema con esta propiedad no permite que un voto pueda ser relacionado con el votante que lo emitió.

2. *Verificación:* Cada votante puede verificar que su voto fue contando correctamente.

3. *Exactitud:* Los votos no pueden ser alterados, duplicados ó eliminados sin que esta acción se detecte. No se permite que votos inválidos sean contados, ni que votos válidos no estén incluidos en los resultados finales.

4. *Democracia:* Sólo pueden emitir su voto las personas que cumplan con los requisitos para hacerlo, dependiendo del tipo y de los requerimientos de la elección que se está llevando a cabo.

5. *Simplicidad:* Los votantes deben ser capaces de terminar el proceso de votación en poco tiempo, en una sesión, y con equipo mínimo.

6. *Flexibilidad:* Un sistema es flexible si permite el uso de una variedad de formatos para la boleta de votación.

7. *Detección de votos duplicados:* El sistema puede detectar si un votante ha emitido dos o más votos y además es capaz de conocer la identidad del votante tramposo. Con esta característica se puede prescindir de un comprobante de votación; ya que si un elector envía dos o más votos, el sistema podrá detectarlos y anularlos.

Una ventaja de las elecciones electrónicas es que, debido a que pueden efectuarse desde cualquier lugar con acceso a la red, es más sencillo participar en éstas. Pues algunas personas no participan en elecciones porque, en el esquema convencional, se necesita que los votantes se trasladen físicamente a las casillas para poder emitir su voto. Otra ventaja, es el hecho de que en principio son más seguras que las convencionales, puesto que se encuentran respaldadas con técnicas criptográficas avanzadas.

Por otro lado, las elecciones vía intranets o vía Internet proporcionan una valor agregado: la privacidad física. Es decir, que mientras en elecciones tradicionales se requiere de la presencia física de la persona que va a emitir su voto, en las elecciones electrónicas esto no es necesario. Lo cual quiere decir, que cualquier persona puede emitir su voto sin necesidad de ser vista por otros. El hecho de que se pueda emitir un voto desde un lugar diferente a la casilla asignada, proporciona además de comodidad, privacidad y seguridad física.

4.2. Esquemas propuestos

La idea de elecciones electrónicas sobre redes de computadoras ha resultado de gran interés y de intenso estudio en aproximadamente los últimos 20 años. Por lo cual, una gran variedad de protocolos criptográficos para este fin ya han sido propuestos.

Estos protocolos pueden clasificarse en:

1) Basados en funciones homomórficas.
2) Basados en firmas a ciegas.

Los protocolos basados en funciones homomórficas [3, 11, 12, 27, 28, 32] requieren en general esquemas complejos de cifrado, pues deben ocultar el contenido de los votos para así preservar la privacidad de los votantes. Detrás de este tipo de sistemas se encuentran técnicas de compartimiento de secretos y pruebas de conocimiento nulo. Incluyen dos procesos: el de cifrado y el de votación. Además, se genera mucho tráfico de información debido a que es necesario transferir todos los votos a través de la red a más de una autoridad. Y aunque este paradigma es muy simple para los votantes, presenta la desventaja de tener una alta complejidad computacional en los procedimientos de conteo.

Los protocolos basados en firmas a ciegas protegen la privacidad del votante ocultando su identidad y dejando el contenido del voto en claro, visible a la autoridad correspondiente. Estos esquemas se conforman de dos fases: la de registro y la de votación.

En 1983 Chaum introdujo el concepto de *firma a ciegas* [1], y sugirió que éstas podrían ser usadas para dar privacidad a elecciones electrónicas. Cinco años después, Chaum propuso un protocolo el cual ocultaba la identidad de los votantes [2]. Sin embargo, las elecciones llevadas a cabo con este esquema podían ser alteradas por un solo elector. Y aunque este protocolo era capaz de detectar dichas alteraciones, éste no podía recuperarse de ellas sin reiniciar las elecciones por completo.

En 1993, Fujioka et al. [6] desarrollaron un esquema práctico de votación que usaba firmas a ciegas para asegurar el anonimato de los votantes. Para hacerlo, cada votante debía cifrar su voto con una llave secreta y mandarlo al centro de conteo a través de un canal anónimo. Sin embargo este esquema presentaba una desventaja, ya que no era simple, pues la fase de votación debía llevarse a cabo en dos fases.

En 1997, L. Cranor y R. Cytron [4] propusieron e implementaron un protocolo basado en el propuesto por Fujioka et al., denominado *Sensus*. La principal diferencia entre estos esquemas era que *Sensus* permitía que los votantes emitieran su voto en una sola sesión, mientras que en el propuesto por Fujioka este proceso debía realizarse en dos sesiones. Sin embargo, en [4] el votante debe mandar su voto cifrado tres veces durante la fase de votación. Esto provoca que el tráfico en la red se incremente, restando eficiencia al protocolo [19]. Además en 1999 Karro – Wang [16] demostraron que *Sensus* es vulnerable

a presentar problemas de colisiones en las llaves utilizadas.

En el esquema de Wen-Sheng [15] et al., se permite que cada votante mande sólo un mensaje anónimo. Con esto se logró disminuir el tráfico en la red. Sin embargo, el problema de la detección de duplicado de votos aún no se resolvía.

En 1998, Mu y Varadharajan [24] propusieron dos esquemas seguros de voto electrónico que, no sólo protegían la privacidad de los votantes, sino que también detectaban el duplicado de votos.

En 1999, Karro – Wang [16] propusieron un esquema seguro para elecciones electrónicas de muy grande escala. Ellos plantearon la posibilidad de utilizar el protocolo HTTPS, para realizar todas las transacciones, en lugar de usar un canal anónimo. Además, se omitía la utilización de firmas a ciegas. Sin embargo, el esquema incluía seis autoridades, lo cual lo hacía ineficiente. Adicionalmente, no poseía robustez, la verificación solo podía hacerse de manera individual, y se permitían múltiples votos por parte de un solo elector, de los cuales sólo el último era considerado para el conteo final [17].

En el 2001, Ray – Narasimhamurthi [25] diseñaron y publicaron un protocolo para voto electrónico anónimo a través de Internet. Incluía tres autoridades y hacía uso de certificados digitales para autenticar al votante, sin embargo presentaba la posibilidad de agregar votos inválidos.

En el 2003, Joaquim, Zúquete y Ferreira [13, 14] implementaron un sistema de votaciones electrónicas (REVS) basado en el trabajo de DuRette [5], el cual mejoraba el sistema EVOX de Herschberg[10]. REVS está implementado en Java y para que un voto sea válido, éste debe contener t firmas de las N entidades Administradoras, donde: $t > \frac{N}{2}$; lo anterior se realiza con el fin de que un sólo elector no pueda enviar más de un voto.

Por último, también en el 2003, Chien et al. y Lin–Hwang–Chang [18] mostraron que en el esquema de Mu y Varadharajan, cualquier votante podía emitir su voto más de una vez sin ser detectado. De esta manera, Lin–Hwang–Chang propusieron entonces un protocolo, basado en [24], que incrementaba la protección contra un posible fraude, mantenía el uso de las firmas a ciegas, y no requería ningún canal especial para votar.

4.3. Esquema de Lin-Hwang-Chang

En el 2003, Lin–Hwang–Chang [18] propusieron una mejora al esquema de Mu y Varadharajan [24], manteniendo el uso de las firmas a ciegas y prescindiendo del uso de un canal especial para votar.

Este protocolo consta de tres fases: autenticación, votación y conteo. Y la notación que utiliza es la siguiente:

- V : nombre del votante
- SA: servidor de autenticación
- SV: servidor de votación
- SC: servidor de conteo
- Cert: certificado digital expedido por una autoridad certificadora
- t : estampa de tiempo
- $\|$: concatenación de bits
- p : número primo largo
- g : un generador para Z_p^*
- $\{e_x, d_x\}, n_x$: un par de llaves RSA para el participante x. Donde $n_x = e_x \times d_x$, y $e_x \times d_x$ mód $\phi(n_x) = 1$

4.3.1. Fase de autenticación

Esta fase consta de 4 pasos:

1. En esta fase el votante debe escoger dos factores de opacidad b_1 y b_2, así como dos números aleatorios k_1 y r, para generar w_1 y w_2.

$$\begin{aligned} w_1 &= g^r b_1^{e_{SA}} \text{ mód } n_{SA} \\ w_2 &= g^{k_1} b_2^{e_{SA}} \text{ mód } n_{SA} \end{aligned} \tag{4.1}$$

   Por último, el votante envía $\{V, SA, Cert_V, t, w_1, w_2, [(w_1\|w_2\|t)^{d_V} \text{ mód } n_V]\}$ al SA.

2. El SA primero verifica la validez del certificado y verifica la firma $[(w_1\|w_2\|t)^{d_V}$ mód $n_V]$. Si la firma es válida, el SA puede estar seguro de que los parámetros que recibió son correctos. Entonces escoge un número aleatorio k_2 y lo almacena en su base de datos relacionándolo con la identidad del votante actual. Por esto, k_2 debe ser único para cada votante. Entonces, SA genera:

$$\begin{aligned} w_3 &= (k_2\|t)^{e_V} \text{ mód } n_V, \\ w_4 &= (w_1 \times SA)^{d_{SA}} \text{ mód } n_{SA} \\ &= (a \times SA)^{d_{SA}} b_1 \text{ mód } n_{SA}, \\ w_5 &= (w_2 \times g^{k_2} \times SA)^{d_{SA}} \text{ mód } n_{SA} \\ &= (y_1 \times SA)^{d_{SA}} b_2 \text{ mód } n_{SA}, \\ w_6 &= (w_2^2 \times g^{k_2} \times SA)^{d_{SA}} \text{ mód } n_{SA} \\ &= (y_2 \times SA)^{d_{SA}} b_2^2 \text{ mód } n_{SA}. \end{aligned} \tag{4.2}$$

donde $a = g^r$, $y_1 = g^{k_1+k_2}$, $y_2 = g^{2k_1+k_2}$. Y así, el SA le responde a V con el siguiente mensaje:

$\{SA, V, w_3, (w_4 \| w_5 \| w_6 \| t)^{e_V} \text{ mód } n_V\}$

3. El votante obtiene k_2 descifrando w_3. De esta manera, V puede calcular y_1 y y_2. Además, V puede calcular las firmas s_1, s_2 y s_3 removiendo los factores de opacidad mediante las siguientes ecuaciones:

$$\begin{aligned} s_1 &= w_4 \times b_1^{-1} = (a \times SA)^{d_{SA}} \text{ mód } n_{SA}, \\ s_2 &= w_5 \times b_2^{-1} = (y_1 \times SA)^{d_{SA}} \text{ mód } n_{SA}, \\ s_3 &= w_6 \times b_2^{-2} = (y_2 \times SA)^{d_{SA}} \text{ mód } n_{SA}. \end{aligned} \tag{4.3}$$

4. El votante aplica el esquema de firma digital ElGamal para firmar el contenido de voto m. Utilizando como llaves públicas y_1 y y_2, y como llaves privadas los valores $x_1 = k_1 + k_2$ y $x_2 = 2k_1 + k_2$ respectivamente. Las dos firmas (a, s_4) y (a, s_5) del voto m pueden ser generadas de la siguiente manera:

$$\begin{aligned} s_4 &= x_1^{-1}(ma - r) \text{ mód } (p-1), \\ s_5 &= x_2^{-1}(ma - r) \text{ mód } (p-1). \end{aligned} \tag{4.4}$$

respectivamente. Y así, V ya puede generar su boleto de votación:

$T = \{s_1 \| s_2 \| s_3 \| s_4 \| s_5 \| a \| y_1 \| y_2 \| m\}$

4.3.2. Fase de votación

1. El votante envía su boleto de votación T al servidor de votación (SV).

2. El SV verifica la validez de a, y_1 y y_2 comparando las siguientes ecuaciones:

$$\begin{aligned} SA \times a &\overset{?}{=} s_1^{e_{SA}} \text{ mód } n_{SA}, \\ SA \times y_1 &\overset{?}{=} s_2^{e_{SA}} \text{ mód } n_{SA}, \\ SA \times y_2 &\overset{?}{=} s_3^{e_{SA}} \text{ mód } n_{SA}. \end{aligned} \tag{4.5}$$

Si las tres firmas anteriores son correctas, el SV procede a verificar las firmas (a, s_4) y (a, s_5) del voto m.

$$g^{ma} \overset{?}{=} y_1^{s_4} a \text{ mód } p$$

$$g^{ma} \overset{?}{=} y_2^{s_5} a \bmod p \tag{4.6}$$

Si estas dos últimas son también correctas, el SV puede estar seguro de que el boleto T es válido. De esta forma, SV almacena todos los boletos de votación válidos que va recibiendo, para después enviarlos en conjunto a través de la red al SC.

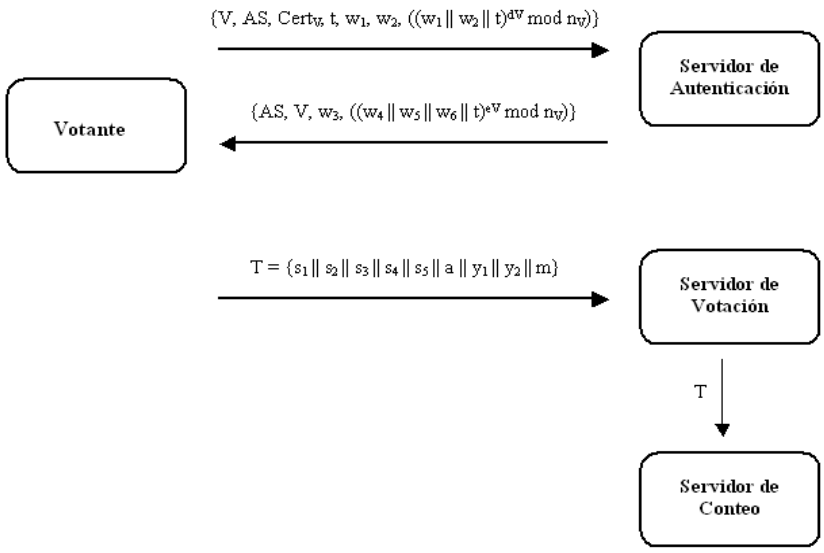

Figura 4.1: Esquema propuesto por Lin–Hwang–Chang

4.3.3. Fase de conteo

Todos los servidores de votación envían sus boletos al SC. El SC publica los boletos y los resultados finales. Aunado a esto, es responsable de detectar los boletos que fueron enviados por un mismo votante. Para realizar lo anterior se debe seguir el siguiente procedimiento:

1. Se asume que un votante utilizó los mismos parámetros y_1, y_2 y a para firmar otro voto m' y envió este nuevo boleto al SV.

2. De esta forma, el SC habrá recibido al menos dos boletos con la forma siguiente:

   $T = \{s_1, s_2, s_3, s_4, s_5, a, y_1, y_2, m\}$

   $T' = \{s_1, s_2, s_3, s'_4, s'_5, a, y_1, y_2, m'\}$

3. De esta manera, SC tiene la habilidad de encontrar la identidad del votante tramposo calculando las siguientes ecuaciones:

$$\begin{aligned} x_1 &= \frac{m'a - ma}{s'_4 - s_4} \text{ mód } (p-1), \\ x_2 &= \frac{m'a - ma}{s'_5 - s_5} \text{ mód } (p-1), \\ k_1 &= x_2 - x_1, \\ k_2 &= x_1 - k_1. \end{aligned} \tag{4.7}$$

4.3.4. Restricciones del esquema de Lin-Hwang-Chang

En esta sección se muestra como el esquema de Lin-Hwang-Chang presenta una restricción, la cual a la hora de implementarse podría generar serios problemas en la comunicación entre el servidor de autenticación y el elector.

El Elector debe elegir 4 números aleatorios en la fase de registro, entre los cuales se encuentra k_1. Y el servidor de autenticación elige un k_2 para cada votante válido. Como puede verse, entre el elector y el SA se generan los dos valores x_1 y x_2, con los cuales, en la fase de votación, se firmará el voto con el esquema de firma digital ElGamal.

$$\begin{aligned} x_1 &= k_1 + k_2, \\ x_2 &= 2k_1 + k_2. \end{aligned} \tag{4.8}$$

Sin embargo, para realizar una firma digital con ElGamal, es necesario que se cumpla la siguiente ecuación:

$$s = k^{-1}(ma - r) \text{ mód } (p-1) \tag{4.9}$$

donde, el valor k necesariamente debe ser primo relativo de $(p-1)$. Y en nuestro caso k será sustituído por los valores x_1 y x_2.

Ahora bien, considerando que el elector elige un valor k_1 arbitrario, y el servidor de autenticación hace lo mismo pero para el valor k_2. Existe la posibilidad de que x_1 y x_2 no sean primos relativos de $(p-1)$. Este hecho no sería del conocimiento del servidor de autenticación debido a que éste nunca llega a conocer k_1; sin embargo, el elector sí lo sabría, pero hasta el final de la fase de autenticación, cuando ya no le sea posible generar sus firmas s_4 y s_5. Y debido a que el SA ya tiene un registro k_2 para el elector en cuestión, a este último no le será posible emitir su voto, ya que el servidor de autenticación no puede asignarle un nuevo k_2.

Por lo anterior, se propone que para firmar el voto, en lugar de utilizar ElGamal, se generen s_4 y s_5 con DSA (*Digital Signature Algorithm*).

Finalmente cabe mencionar que en este esquema, Lin *et al.* proponen que, durante la fase de autenticación, el SA le responda al Votante el siguiente mensaje:

$$\{SA, V, w_3, (w_4 \| w_5 \| w_6 \| t)^{e_V} \text{ mód } n_V\}$$

Sin embargo, a partir del mensaje anterior es imposible que el Votante pueda descifrar w_4, w_5 y w_6. Ya que estos valores son de tamaño n_{SA}, es decir de 1024 bits aproximadamente; y concatenados forman un número de aproximadamente 3072 bits. Por lo tanto, este último número no puede cifrarse con una llave pública RSA de 1024 bits, porque una restricción de este algoritmo es que el proceso de cifrado no puede aplicarse a valores de mayor tamaño que el módulo; y en este caso el módulo es n_V de 1024 bits.

Por lo anterior, en el esquema propuesto se ha modificado la estructura del mensaje de respuesta del SA al V, durante la fase de autenticación.

4.4. Esquema implementado

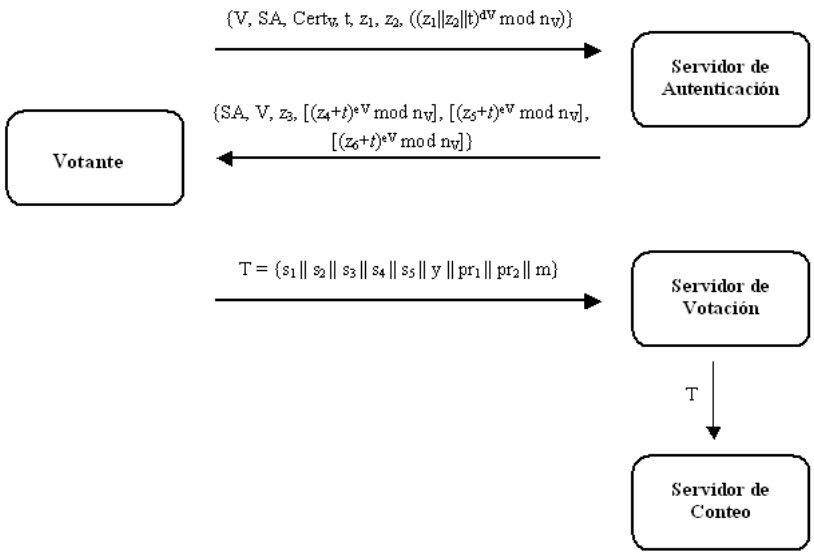

Figura 4.2: Esquema implementado en SELES

El protocolo que se implementó en SELES está basado en el de Lin–Hwang–Chang. Sin embargo se le hicieron algunas modificaciones para eliminar las restricciones detectadas en [18].

Si se sustituye el uso de ElGamal por DSA, se tienen que realizar algunos ajustes al protocolo de votación. Ya que con ElGamal se utiliza aritmética modular mód p y mód

$(p-1)$, mientras que DSA requiere aritmética modular mód p y mód q. Y así, con esta modificación al protocolo original se garantiza que, no importando los valores k_1 y k_2 que elijan el votante y el servidor de autenticación respectivamente, el voto pueda ser firmado correctamente antes de ser enviado al servidor de votación.

Por lo tanto el protocolo que se propone a continuación también consta de tres fases, y la notación que utiliza es la siguiente:

- V : votante
- SA: sevidor de autenticación
- SV: servidor de votación
- SC: servidor de conteo
- q : parámetro de DSA, $2^{159} < q < 2^{160}$
- p : dado un l tal que $0 \leq l \leq 8$, p debe cumplir, $2^{511+64l} < p < 2^{512+64l}$, con la propiedad de que q divide a $(p-1)$
- g : un generador para Z_p^*
- a : llave privada de DSA $1 \leq a \leq q-1$
- $\alpha = g^{(p-1)/q}$ mód p
- $y = \alpha^a$ mód p
- Cert: certificado digital expedido por una autoridad certificadora
- $\{e_x, d_x\}, n_x$: un par de llaves RSA para el participante x. Donde $n_x = e_x \times d_x$, y $e_x \times d_x$ mód $\phi(n_x) = 1$

4.4.1. Fase de autenticación

Esta sección consta de 3 pasos:

1. El votante elige dos factores de opacidad b_1 y b_2, y dos números aleatorios k_1 y a. Con estos datos y con los parámetros de DSA se generan los valores y, z_1 y z_2 de la siguiente manera: $y = \alpha^a$ mód p,

$$\begin{aligned} z_1 &= [(\alpha^a \text{ mód } p) \times (b_1^{e_{SA}})] \text{ mód } n_{SA}, \\ z_2 &= [(\alpha^{k_1} \text{ mód } p) \times (b_2^{e_{SA}})] \text{ mód } n_{SA}. \end{aligned} \tag{4.10}$$

   donde p y α son parámetros públicos.

   Y así el votante envía $\{V, SA, Cert_V, t, z_1, z_2, [(z_1 \| z_2 \| t)^{d_V} \text{ mód } n_V]\}$ al SA.

2. El SA comprueba la validez de la identidad de V verificando la firma recibida $[(z_1\|z_2\|t)^{d_V} \text{ mód } n_V]$ con la llave pública contenida en $Cert_V$. Si es una firma válida, entonces SA escoge un número aleatorio k_2 y lo almacena en su base de datos como identificador del V actual. Por lo cual, el valor k_2 debe ser único para cada votante. Después SA genera z_3, z_4, z_5 y z_6 como se muestra a continuación:

$$\begin{aligned} z_3 &= (k_2\|t)^{e_V} \text{ mód } n_V, \\ z_4 &= (z_1 \times SA)^{d_{SA}} \text{ mód } n_{SA} \\ &= [(\alpha^a \text{ mód } p) \times SA]^{d_{SA}} b_1 \text{ mód } n_{SA}, \\ z_5 &= (z_2 \times (\alpha^{k_2} \text{ mód } p) \times SA)^{d_{SA}} \text{ mód } n_{SA} \\ &= [(\alpha^{k_1} \text{ mód } p) \times (\alpha^{k_2} \text{ mód } p) \times SA]^{d_{SA}} b_2 \text{ mód } n_{SA}, \\ z_6 &= (z_2^2 \times (\alpha^{k_2} \text{ mód } p) \times SA)^{d_{SA}} \text{ mód } n_{SA} \\ &= [(\alpha^{2k_1} \text{ mód } p) \times (\alpha^{k_2} \text{ mód } p) \times SA]^{d_{SA}} b_2^2 \text{ mód } n_{SA}. \end{aligned} \tag{4.11}$$

Finalmente SA le envía el siguiente mensaje de respuesta a V:

$$\{SA, V, z_3, [(z_4 + t)^{e_V} \text{ mód } n_V], [(z_5 + t)^{e_V} \text{ mód } n_V], [(z_6 + t)^{e_V} \text{ mód } n_V]\}$$

Como puede verse en este mensaje de respuesta, los valores de z_4, z_5 y z_6 están cifrados de manera separada con la llave pública del votante, además de que previamente se les ha sumando la estampa de tiempo t. Esto ha sido debido a que no es posible cifrar los tres valores concatenados (con una longitud de aproximadamente 3072 bits), ya que el tamaño de n_V es de 1024 bits.

3. El votante descifra z_3 para obtener el valor de k_2. También descifra los valores z_4, z_5 y z_6 elevando a d_V los tres últimos datos del mensaje de respuesta, aplicándoles módn_V, y restándoles el valor t.

Después se remueven los factores de opacidad, generando así las firmas s_1, s_2 y s_3.

$$\begin{aligned} s_1 &= z_4 \times b_1^{-1} = [(\alpha^a \text{ mód } p) \times SA]^{d_{SA}} \text{ mód } n_{SA}, \\ s_2 &= z_5 \times b_2^{-1} = [(\alpha^{k_1} \text{ mód } p) \times (\alpha^{k_2} \text{ mód } p) \times SA]^{d_{SA}} \text{ mód } n_{SA}, \\ s_3 &= z_6 \times b_2^{-2} = [(\alpha^{2k_1} \text{ mód } p) \times (\alpha^{k_2} \text{ mód } p) \times SA]^{d_{SA}} \text{ mód } n_{SA}. \end{aligned} \tag{4.12}$$

4.4.2. Fase de votación

1. En la fase de votación, el votante procede a firmar el voto que ha elegido (m) con DSA. Para este procedimiento se utilizarán como llaves privadas a x_1 y x_2, valores que el elector puede generar sin ningún problema, ya que previamente ha descifrado

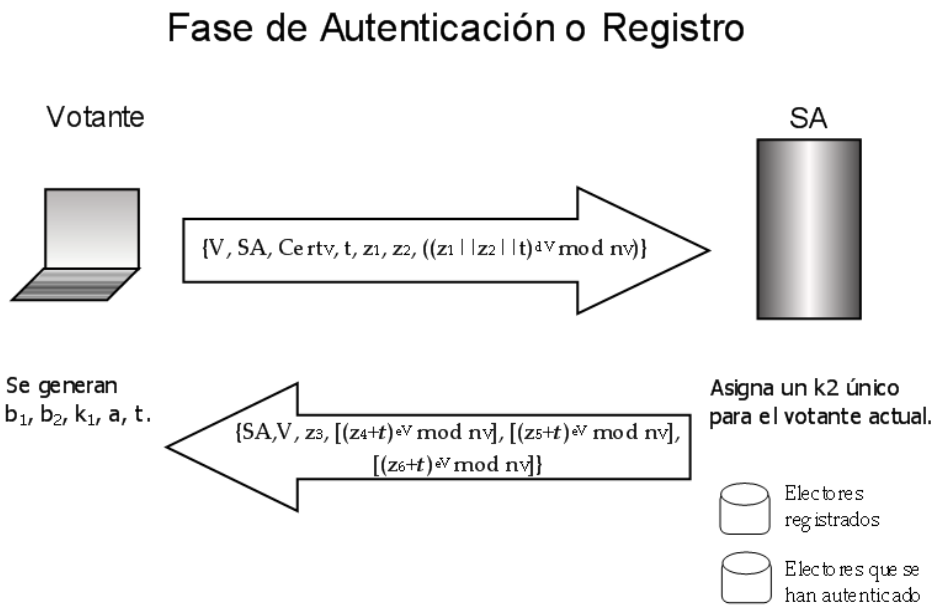

Figura 4.3: Primera fase del esquema propuesto

a k_2. Y las correspondientes llaves públicas serán r_1 y r_2.

$$\begin{aligned} x_1 &= k_1 + k_2, \\ x_2 &= 2k_1 + k_2, \\ r_1 &= (\alpha^{k_1+k_2} \bmod p) \bmod q, \\ r_2 &= (\alpha^{2k_1+k_2} \bmod p) \bmod q. \end{aligned} \tag{4.13}$$

De esta manera, se pueden generar dos firmas (r_1, s_4) y (r_2, s_5), con las siguientes ecuaciones:

$$\begin{aligned} s_4 &= x_1^{-1}(m + ar_1) \bmod q, \\ s_5 &= x_2^{-1}(m + ar_2) \bmod q. \end{aligned} \tag{4.14}$$

Para finalizar, se obtienen los valores l_1, l_2, pr_1 y pr_2.

$$\begin{aligned} l_1 &= [((\alpha^{k_1} \bmod p) \bmod n_{SA}) \times \\ &\qquad ((\alpha^{k_2} \bmod p) \bmod n_{SA})] \bmod n_{SA}, \\ l_2 &= [((\alpha^{k_1} \bmod p)^2 \bmod n_{SA}) \times \\ &\qquad ((\alpha^{k_2} \bmod p) \bmod n_{SA})] \bmod n_{SA}, \\ pr_1 &= [(r_1 \times n_{SA}) + (l_1 \times q)] \bmod (n_{SA} \times q), \\ pr_2 &= [(r_2 \times n_{SA}) + (l_2 \times q)] \bmod (n_{SA} \times q). \end{aligned} \tag{4.15}$$

Estos dos últimos valores encapsulan $(r_1 \& l_1)$ y $(r_2 \& l_2)$ mediante el *Teorema chino del residuo*. Esto es con la finalidad de que se puedan realizar las verificaciones correspondientes utilizando los módulos n_{SA} y q. Ya que el SV necesita verificar:

- 3 firmas con módulo n_{SA}, las cuales son las firmas que el SA le proporcionó al votante,
- 2 firmas con módulo q, correspondientes a las firmas del voto (m) con DSA.

Por último se genera el boleto de votación de la siguiente forma:

$B = \{s_1, s_2, s_3, s_4, s_5, y, pr_1, pr_2, m\}$

2. El elector V envía su boleto de votación al SV. El SV ejecuta las 5 verificaciones de las firmas para comprobar la validez del boleto. Las primeras 3 ecuaciones de verificación son las siguientes:

$$\begin{aligned} (SA \times y) \bmod n_{SA} &\stackrel{?}{=} s_1^{e_{SA}} \bmod n_{SA}, \\ (SA \times \frac{pr_1}{q}) \bmod n_{SA} &\stackrel{?}{=} s_2^{e_{SA}} \bmod n_{SA}, \\ (SA \times \frac{pr_2}{q}) \bmod n_{SA} &\stackrel{?}{=} s_3^{e_{SA}} \bmod n_{SA}. \end{aligned} \tag{4.16}$$

Ahora bien, para verificar las firmas hechas con DSA, primero se deben obtener los valores r_1 y r_2 de la siguiente manera:

$$\begin{aligned} r_1 &= \frac{pr_1}{n_{SA}} \bmod q, \\ r_2 &= \frac{pr_2}{n_{SA}} \bmod q. \end{aligned} \tag{4.17}$$

Una vez hecho lo anterior, se procede a verificar las firmas mediante el procedimiento *verificarDSA* (ver Algoritmo 1).

Algoritmo 1 (*verificarDSA*) [4.1]

ENTRADA:
valores r, s
SALIDA:
valor v

1. $w = s^{-1}$ mód q
2. $u_1 = w \cdot m$ mód q
3. $u_2 = r \cdot w$ mód q
4. $v = (\alpha^{u_1} y^{u_2}$ mód $p)$ mód q
5. return v

De esta manera, para poder aceptar el boleto, sólo resta verificar que las dos últimas firmas s_4 y s_5 sean válidas:

$$r_1 \stackrel{?}{=} verificarDSA(r_1, s_4),$$

$$r_2 \stackrel{?}{=} verificarDSA(r_2, s_5). \tag{4.18}$$

3. Si las cinco firmas resultaron ser correctas, entonces el SV acepta el boleto como válido y lo almacena. Una vez que se ha terminado el tiempo establecido para la emisión de votos, el SV procede a enviar juntos todos los boletos válidos que recibió al SC a través de la red.

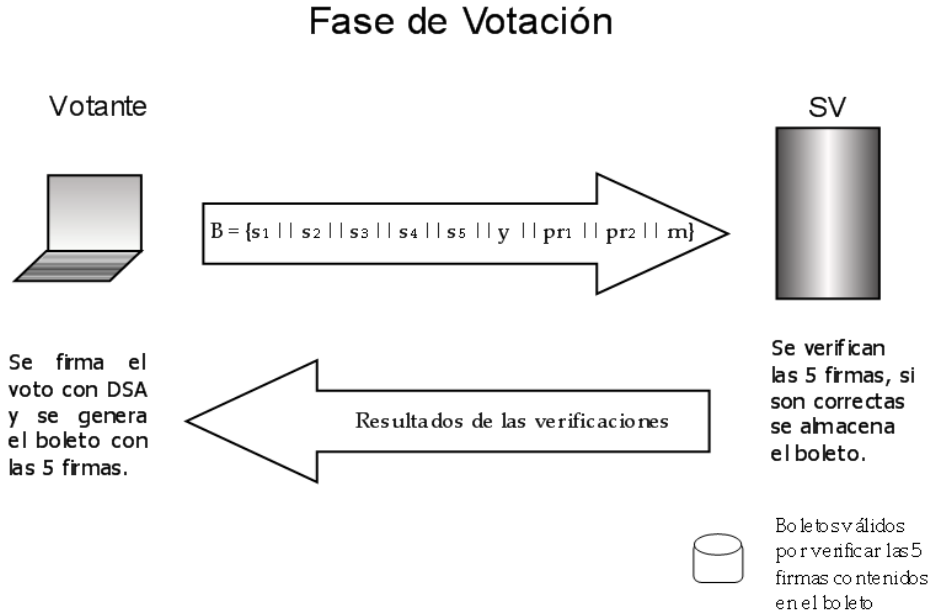

Figura 4.4: Segunda fase del esquema propuesto

4.4.3. Fase de conteo

El SC debe recibir los boletos válidos de todos los servidores de votación y contar una sola vez los boletos que sean idénticos. De esta manera, el SC realiza la cuenta final y publica todos los boletos válidos.

En esta fase es posible detectar a un elector que haya enviado dos o más boletos con diferente voto. Para ello, se debe considerar que un votante utiliza las mismas llaves para firmar votos diferentes. Entonces, el SC recibiría al menos dos boletos con la forma siguiente:

$B_1 = \{s_1, s_2, s_3, s_4, s_5, y, pr_1, pr_2, m\}$
$B_2 = \{s_1, s_2, s_3, s'_4, s'_5, y, pr_1, pr_2, m'\}$

Con la información de estos dos boletos, el SC es capaz de identificar al votante que los emitió. Esto lo hace mediante las ecuaciones:

$$\begin{aligned} x_1 &= \frac{m' - m}{s'_4 - s_4} \text{ mód } q, \\ x_2 &= \frac{m' - m}{s'_5 - s_5} \text{ mód } q, \\ k_1 &= x_2 - x_1, \\ k_2 &= x_1 - k_1. \end{aligned} \tag{4.19}$$

Y como se recordará, todos los valores k_2 asignados a cada elector se encuentran almacenados en la base del SA. De esta forma el SC sólo tiene que solicitarle al SA el nombre del elector al cual está asociado el valor k_2 que ha obtenido, para así conocer la identidad del votante tramposo.

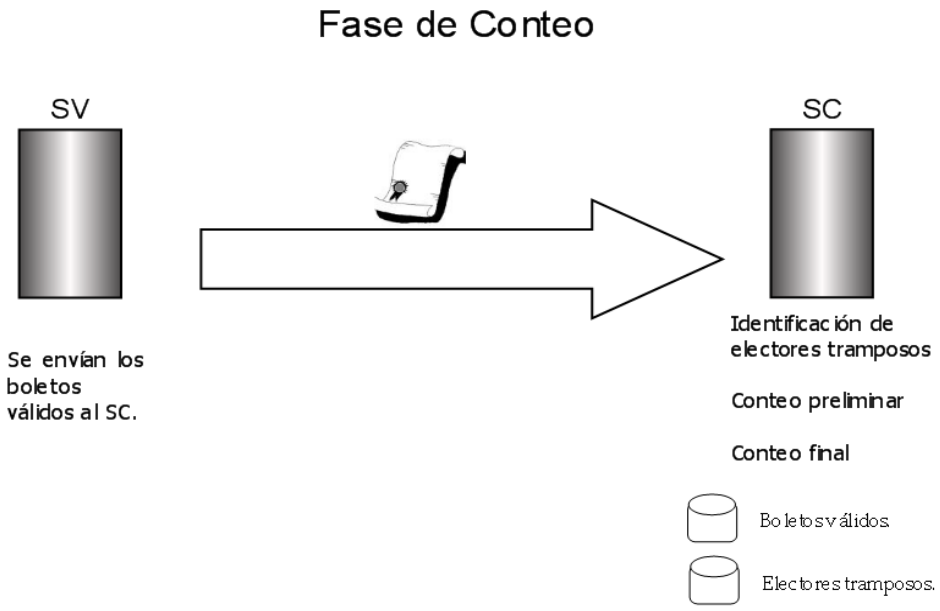

Figura 4.5: Tercera fase del esquema propuesto

Capítulo 5

Diseño e implementación

A través de los capítulos anteriores, se ha afirmado que las elecciones electrónicas pueden ser una alternativa viable a las elecciones convencionales. Asimismo, las primeras pueden ofrecer propiedades adicionales deseables, tales como la seguridad y la privacidad física. Sin embargo, se debe contar con las medidas de seguridad pertinentes para que estos sistemas no sean comprometidos, causando así resultados electorales fraudulentos o violaciones a la privacidad de los electores participantes.

Por lo anterior, en este trabajo de tesis, se ha diseñado e implementado un sistema para elecciones electrónicas seguras (SELES) a mediana escala. El cual hace uso de la tecnología *Cliente–Servidor* en Internet, y de diversas herramientas y técnicas criptográficas que aseguran su correcto funcionamiento.

En este capítulo se describe el diseño y la implementación de SELES. Por lo cual, en la sección 5.1 se presenta una descripción de la arquitectura general del sistema. Después, en la sección 5.2 se exponen las características de diseño e implementación de las *autoridades* involucradas en nuestro protocolo de votación. En la sección 5.3 se detalla el diseño y desarrollo de la *aplicación elector*. Y por último, en la sección 5.4 se muestran los detalles de implementación.

5.1. Arquitectura del sistema

Como se ha mencionado anteriormente, SELES es un Sistema para Elecciones Electrónicas Seguras; el cual, a través del uso de diversas herramientas criptográficas y de tecnología computacional, provee características deseables para un sistema de votación.

Más concretamente, SELES utiliza criptografía de llave pública, certificados digitales, firmas a ciegas con RSA, esquema de firma digital DSA, estampas de tiempo, el teorema chino del residuo, y funciones *hash*, entre otros.

SELES ha sido diseñado para que la emisión de votos se haga a través de Internet, ya que las *autoridades* correspondientes están implementadas utilizando el modelo de programación *Cliente–Servidor*.

Para realizar las acciones propias del votante, se ha diseñado y desarrollado una *aplicación elector*, la cual puede ejecutarse desde una computadora personal (PC) o una computadora portátil (lap-top), o incluso un asistente personal digital (PDA).

Además debido a que los votos se emiten en línea, los votantes participantes en una determinada elección, pueden tener una conexión alámbrica o inalámbrica a Internet (ver figura 5.1).

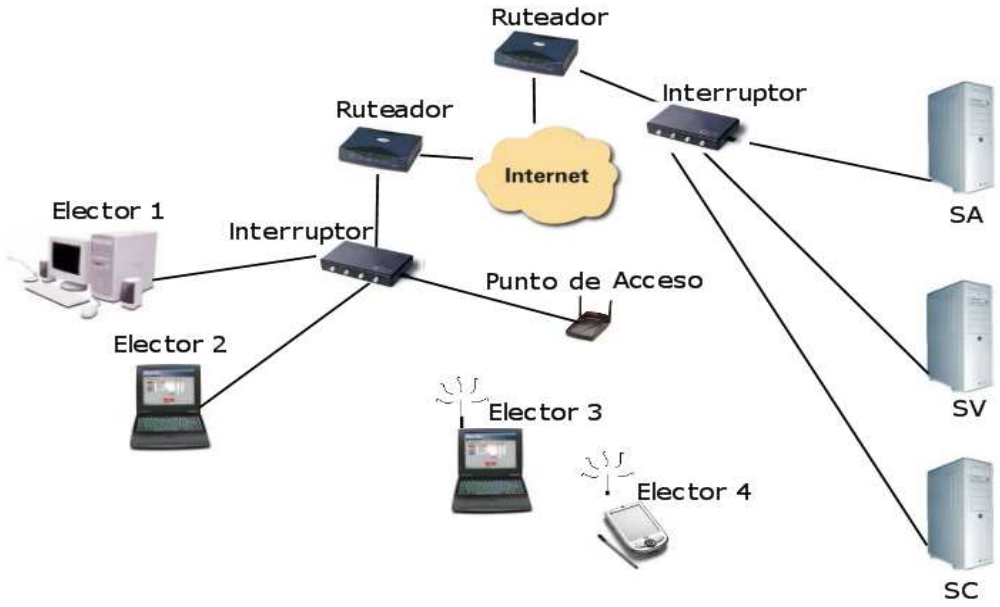

Figura 5.1: Arquitectura general

Por lo cual, SELES puede ejecutarse de manera adecuada en:

1. Una intranet alámbrica, conectada a través de un interruptor (*switch*), con acceso a Internet, cuyos nodos incluirían computadoras personales y computadoras portátiles.

2. Una intranet inalámbrica, conectada a través de un punto de acceso, con conexión a internet; conformada por nodos que podrían ser: computadoras personales y portátiles equipadas con tarjeta de red inalámbrica, y dispositivos móviles, específicamente asistentes personales digitales (PDA's).

Los electores que voten a través de una computadora con gran capacidad (PC o laptop), podrán hacerlo sin necesidad de instalar la *aplicación elector* en su máquinas. Lo anterior se debe a que todas las acciones, que requiera hacer el votante, se realizan a través de un *applet firmado*, el cual se descarga en la fase de registro del proceso de votación.

En el caso de los electores que deseen emitir su voto desde un asistente personal digital, éstos sí tendrán que instalar en sus dispositivos la *aplicación elector* de manera previa; puesto que actualmente sólo algunos navegadores para PDA's son capaces de soportar *applets*, y casi ninguno acepta *applets firmados*.

Debido a que los asistentes personales digitales son dispositivos móviles con capacidad limitada, las capas de desarrollo del sistema requieren consideraciones diferentes respecto a las de las computadoras de escritorio y/o portátiles.

En la figura 5.2 se muestran las capas de desarrollo que se utilizaron para implementar cada una de las entidades involucradas en nuestro sistema.

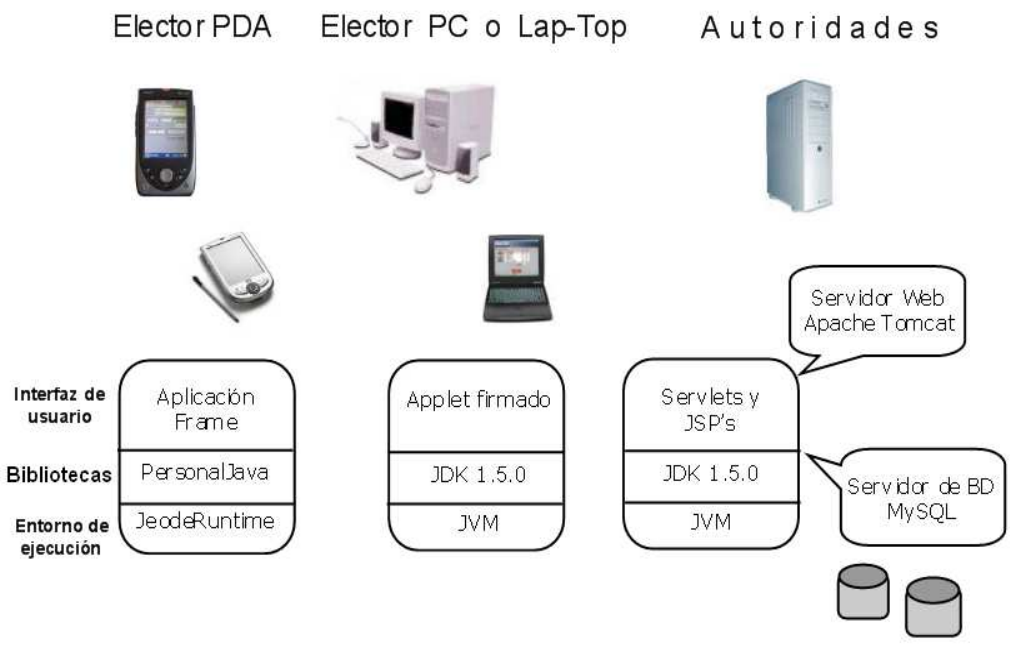

Figura 5.2: Capas de desarrollo para cada entidad

Los detalles de esta figura se presentarán en la sección 5.4.

De manera específica, la arquitectura del sistema para el proceso de votación se muestra en la figura 5.3. Como se puede apreciar, para poder emitir un voto se debe interactuar

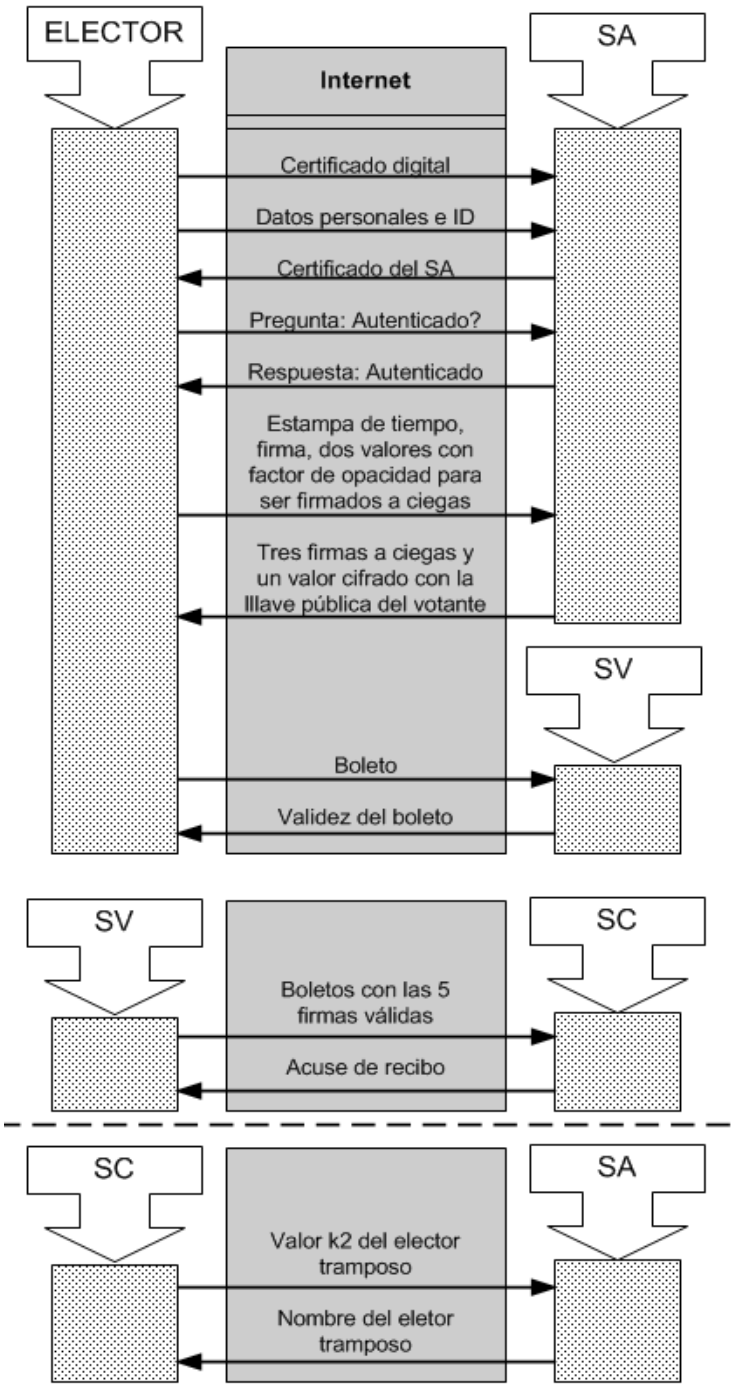

Figura 5.3: Arquitectura de SELES

con SA y con SV en las fases de autenticación y votación respectivamente. En la fase de conteo SC recibe los boletos de votación válidos. Y para identificar a los electores que

enviaron más de un voto, SC interactúa con SA.

5.2. Autoridades

En esta sección se exponen las características de diseño e implementación de cada una de las autoridades (*servidor de autenticación, servidor de votación y servidor de conteo*) existentes en el protocolo de votación de SELES.

La función general del servidor de autenticación es la de verificar que el elector sea quien dice ser, y firmar a ciegas las llaves con las que, posteriormente, el elector firmará su voto. En otras palabras, es el encargado del empadronamiento.

El servidor de votación recibe los votos firmados adecuadamente con llaves autorizadas por el servidor de autenticación, es decir, posee la funcionalidad de una urna electoral.

La tercera autoridad, es decir, el servidor de conteo se encarga de recolectar todos los votos, realizar el conteo final y detectar los votos emitidos por un solo elector.

Sin embargo, existe también una cuarta autoridad, la *autoridad certificadora* **ACERPAM**. Ésta es la que se encarga de expedir, previamente al proceso de votación, los certificados digitales de los votantes y sus respectivas llaves privadas cifradas con las contraseñas que ellos mismos eligen. Para más detalles respecto a la autoridad certificadora utilizada en este trabajo véase apéndice B.

5.2.1. Servidor de autenticación

El Servidor de autenticación (SA), como su nombre lo indica, es el encargado autenticar a los votantes que estén registrados correctamente en la base de datos del padrón para determinadas elecciones.

En la figura 5.4 se puede ver el digrama de flujo para el SA.

Como puede verse en este diagrama de flujo, el SA es quien recibe el certificado digital y los datos personales del votante, es decir, se encarga de registrarlo. Después, verifica el ID y la firma que ha recibido. A continuación revisa si el elector actual no ha sido previamente autenticado, es decir, si no se le ha firmado a ciegas con anterioridad. Si esto último sucede, se procede a leer de la base de datos el valor k_2 que ya se le ha asignado a dicho elector. Sin embargo, si es la primera vez que el votante contacta al SA, entonces se le genera un valor k_2, para identificarlo de manera única en la base de datos. Finalmente, SA firma a ciegas los valores correspondientes, de acuerdo al protocolo de comunicación (ver sección 4.4); cifra las firmas y el valor k_2 junto con la estampa de tiempo; y genera

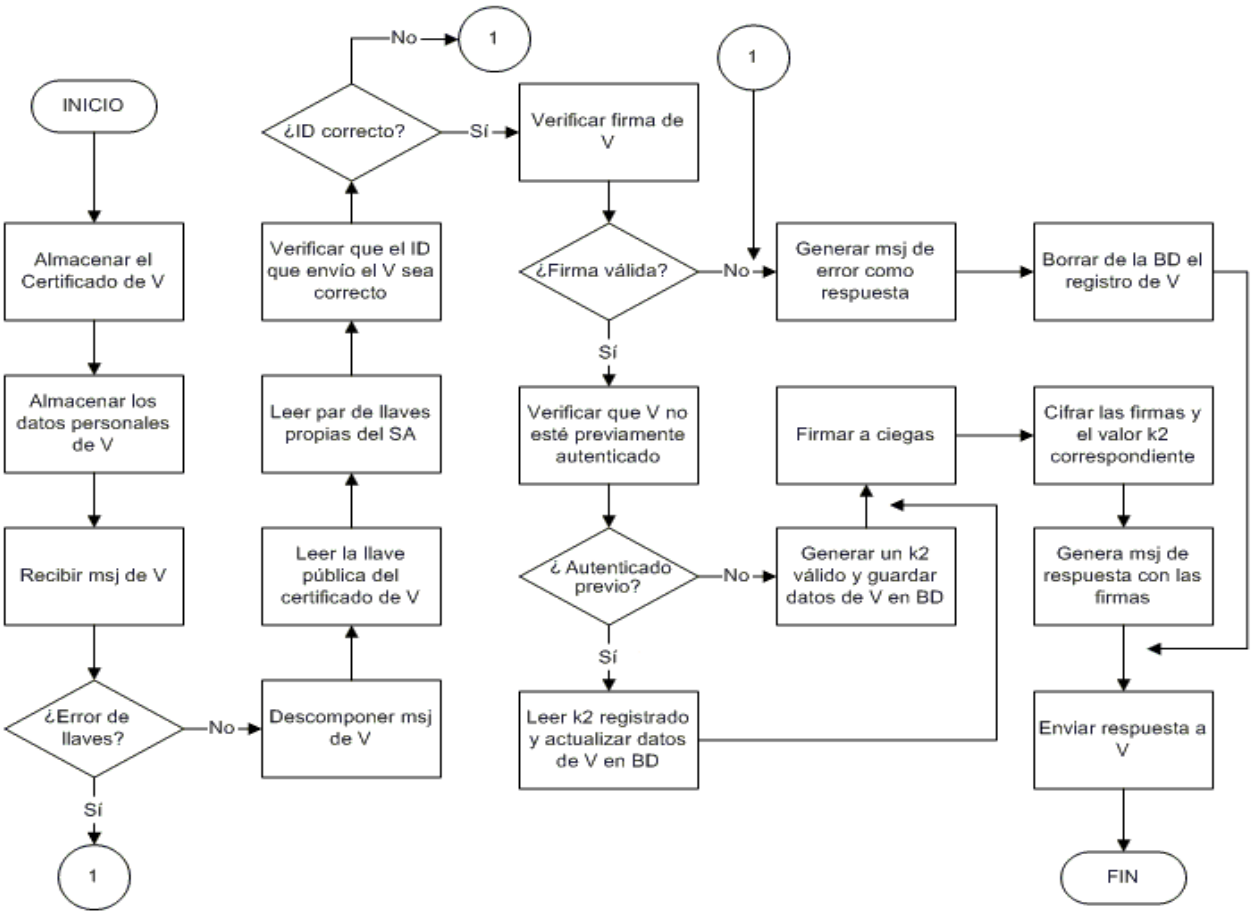

Figura 5.4: Diagrama de flujo del SA

el mensaje de respuesta, que posteriormente le enviará al elector.

Para poder ver la interacción entre el SA y el votante de una manera más específica, se presenta a continuación el correspondiente diagrama de secuencia.

El diagrama 5.6 muestra la composición de las diferentes clases que se desarrollaron para darle al SA la funcionalidad requerida.

Como se puede observar las clases principales son servlets, los cuales hacen uso de las demás clases para llevar a cabo su función:

- *ServletSA*: Es el servlet principal, y es el encargado de autenticar al votante y generar el mensaje de respuesta con las firmas a ciegas y el valor k_2 asignado.
- *ServletCertSA*: Envía el certificado del SA al votante una vez que ha aceptado el applet.
- *ServletIdentificador*: Se encarga de recibir, por parte del SC, los identificadores ($k_2's$)

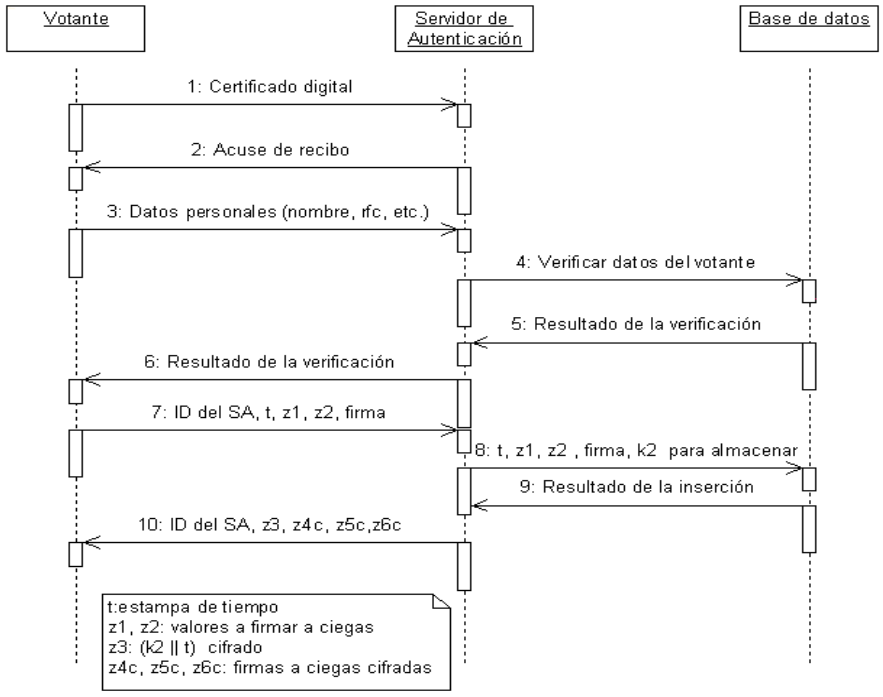

Figura 5.5: Diagrama de secuencia entre el votante y el SA

de los electores tramposos, y de enviarle a cambio los nombres de dichos electores, los cuales SA tiene registrados en su base de datos.

- *ServlertRegPDA*: Recibe y almacena los datos personales de un elector que específicamente utiliza un asistente personal digital.
- *ServletCertPDA*: Recibe y almacena el certificado digital que envía un votante desde un asistente personal digital.
- *CertNo*: Esta clase se encarga de leer el valor del exponente y del módulo de la llave pública contenida en un determinado certificado digital.
- *ParametrosDSA*: Contiene los parámetros DSA que se utilizarán en el protocolo de comunicación.
- *ParamBD*: Contiene los parámetros que requiere cada autoridad para realizar una conexión a su respectiva base de datos.
- *Pstore*: Se hace cargo de descifrar y leer una llave privada, indicando la contraseña correcta.

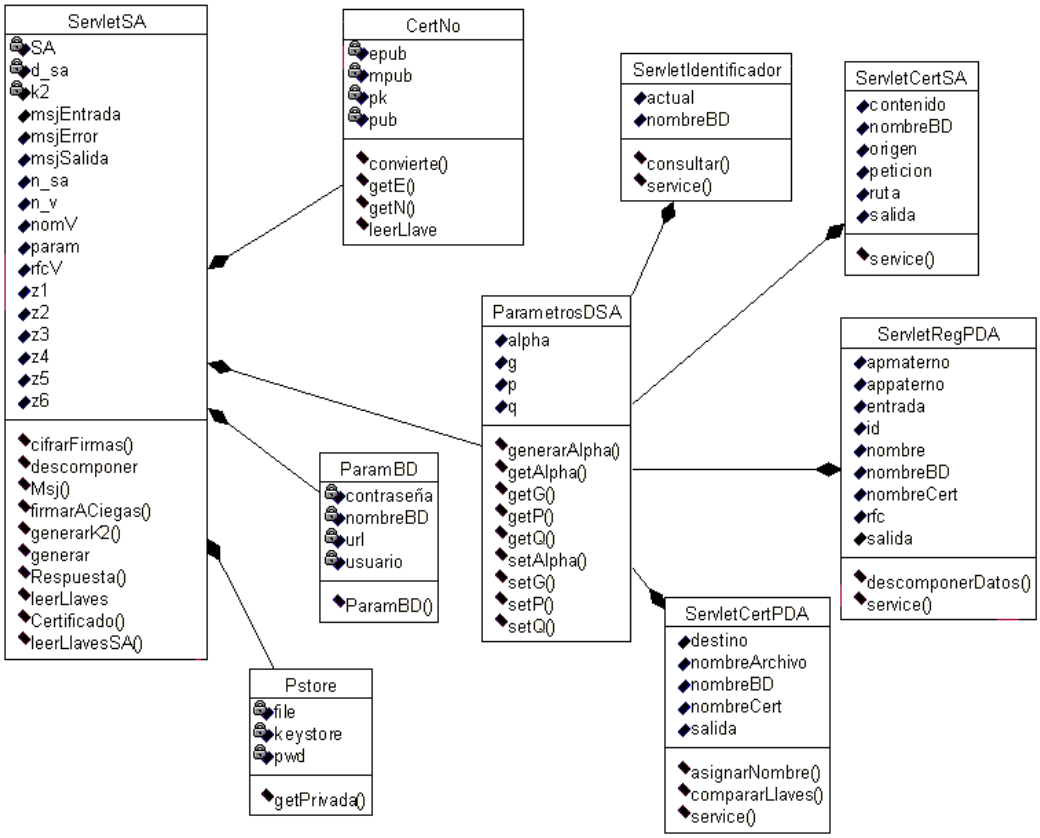

Figura 5.6: Diagrama de clases para el SA

5.2.2. Servidor de votación

El servidor de votación (SV) tiene la función de recibir los boletos por parte de todos los electores participantes en determinadas elecciones. Los boletos que se reciben contienen:

1. el *voto* elegido por su respectivo emisor,
2. los *parámetros públicos* para verificar las 2 firmas que se hicieron al voto, y
3. las *5 firmas* a verificar; tres con RSA y dos con DSA.

En la figura 5.7 se muestra el digrama de flujo del SV. En esta figura claramente se señala que SV recibe primero el boleto. Después, obtiene la llave pública del SA, ya que las tres firmas RSA deben ser verificadas con esta última. Una vez leída la llave de SA, se procede a verificar las tres firmas hechas RSA. A continuación se verifican las generadas con DSA. Si las 5 firmas son correctas, se genera el mensaje de respuesta para el elector indicándole que su boleto es correcto. Sin embargo, si alguna firma resultó inválida, el mensaje de respuesta avisa al votante que su boleto ha sido rechazado.

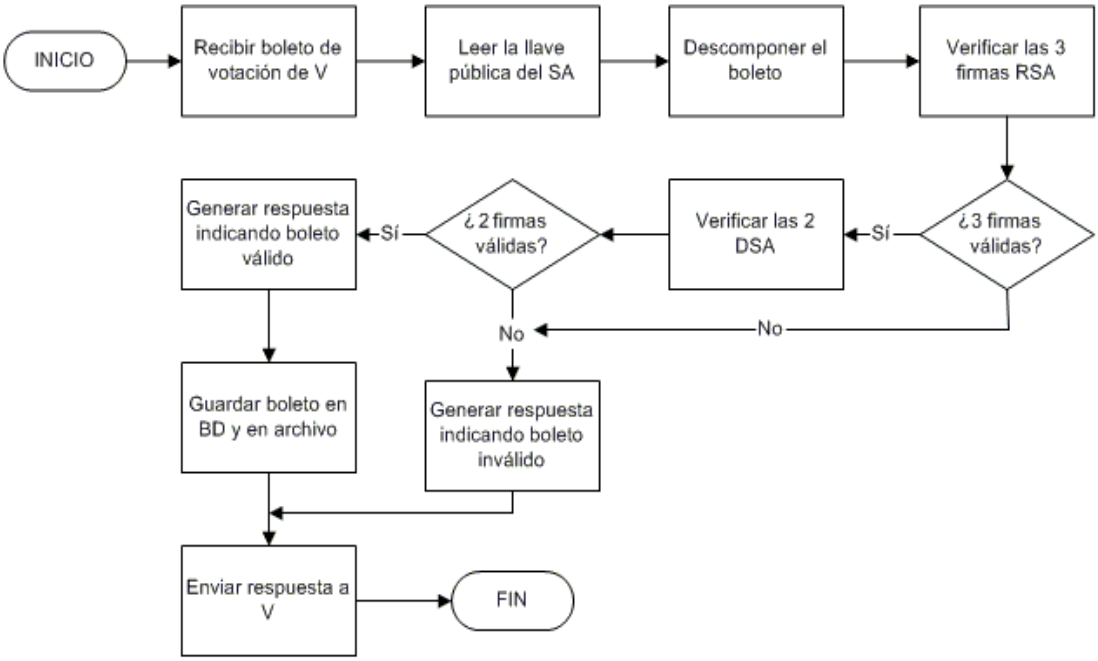

Figura 5.7: Diagrama de flujo del SV

Para observar la interacción más específica entre el SV y el elector participante, se expone en la figura 5.8 el diagrama de secuencia que involucra a estas dos entidades. En la figura 5.9 se presenta el diagrama de clases de esta autoridad. La clase *ParametrosDSA* y *ParamBD* se han descrito previamente en la subsección 5.2.1. Por lo tanto a continuación, sólo se describe la tercera clase del diagrama:

- *ServletSV*: Este servlet se encarga de recibir los boletos de votación de los electores, y de verificar las cinco firmas contenidas en los mismos.

5.2.3. Servidor de conteo

El servidor de conteo (SC) se hace cargo de recibir, del servidor de votación, todos los boletos registrados con las 5 firmas válidas. Además, tiene la responsabilidad de contar los votos de manera exacta para su posterior publicación.

El diagrama de flujo de datos del SC se presenta en la figura 5.10. En esta figura puede apreciarse como SC, en un principio, recibe y almacena los boletos. Después, se encarga de buscar y marcar aquellos boletos que resulten idénticos (con el fin de considerarlos una sola vez en el conteo final), y de hacer lo mismo con los que sean sospechosos de haber sido enviados por un mismo elector. En caso de que se encuentren boletos sospechosos, SC

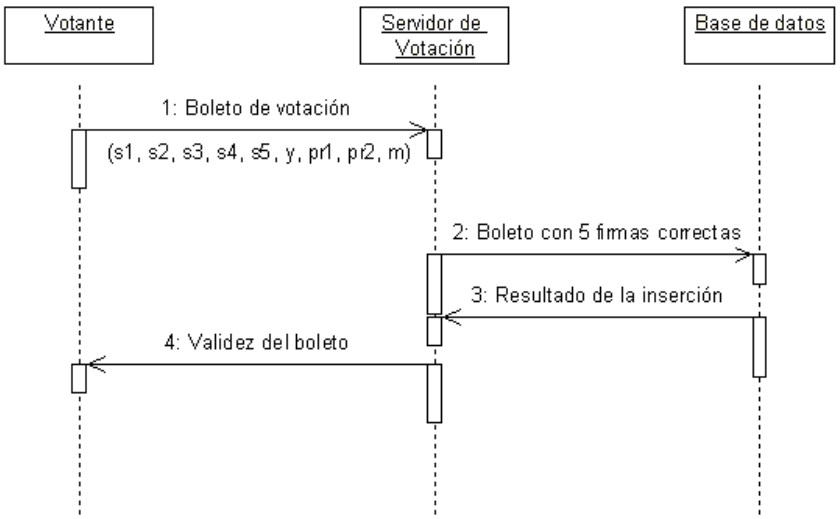

Figura 5.8: Diagrama de secuencia entre el votante y el SV

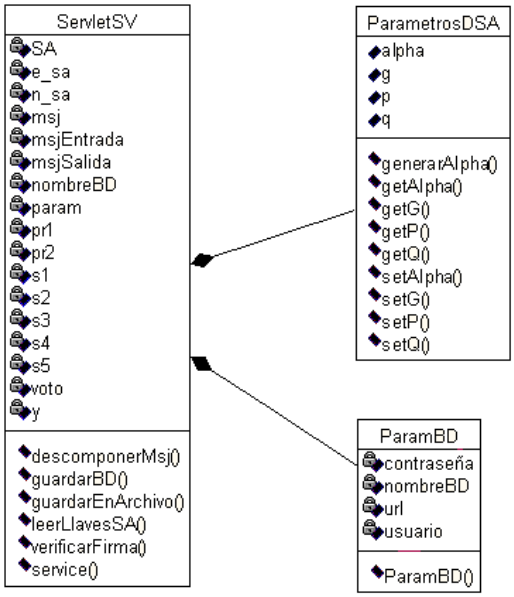

Figura 5.9: Diagrama de clases para el SV

debe obtener sus identificadores para, posteriormente, enviárselos a SA con el objetivo de que este último le regrese los nombres de los electores tramposos correspondientes a tales identificadores. Finalmente SC realiza el conteo final y publica los resultados pertinentes.

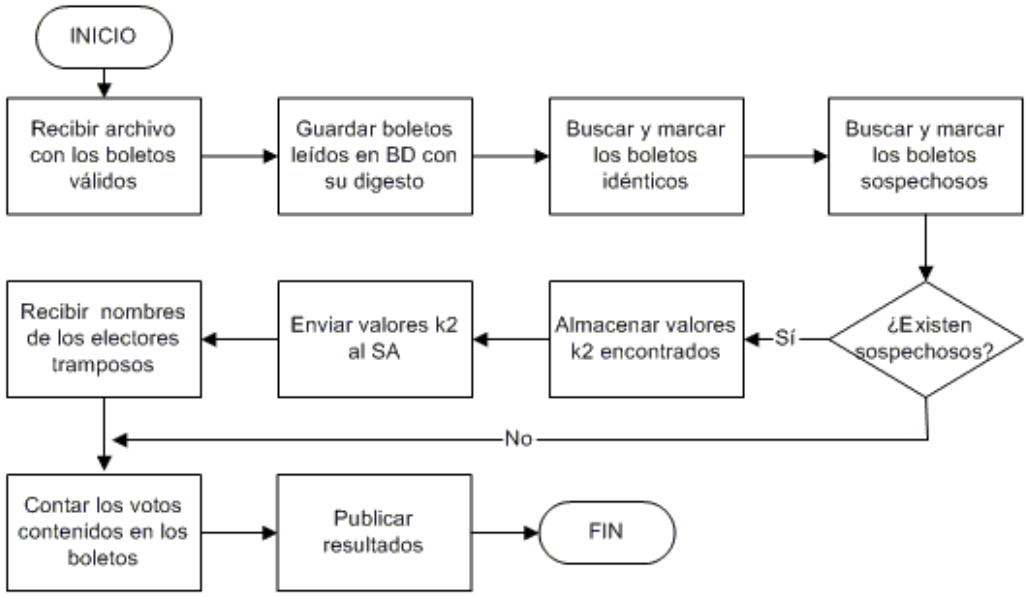

Figura 5.10: Diagrama de flujo del SC

También se muestra, en la figura 5.11, el diagrama de secuencia entre el SC y el SV. En este diagrama se pueden ver más detalladamente los mensajes que son intercambiados entre estas dos autoridades.

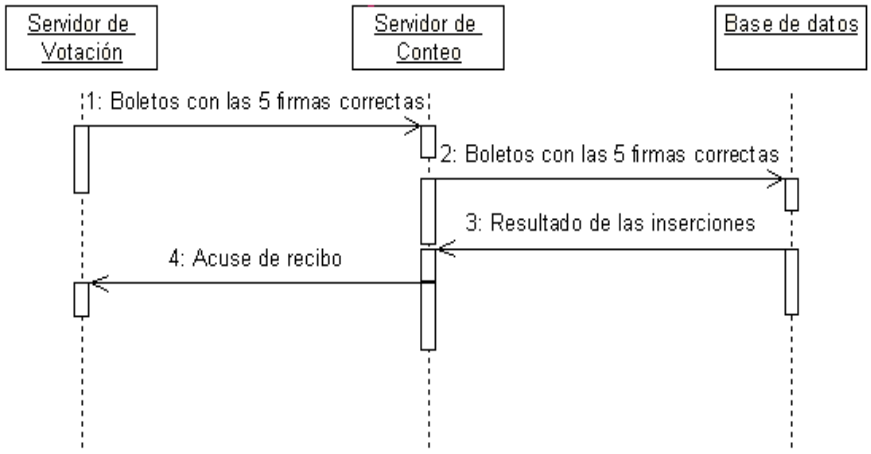

Figura 5.11: Diagrama de secuencia entre el SV y el SC

Otra interacción importante es la que se lleva a cabo entre el SC y el SA. La cual es necesaria debido a que SC identifica a los votantes tramposos con la ayuda de SA, el

cual tiene toda la información acerca de los electores que participaron en el proceso de votación. El diagrama correspondiente puede verse en la figura 5.12.

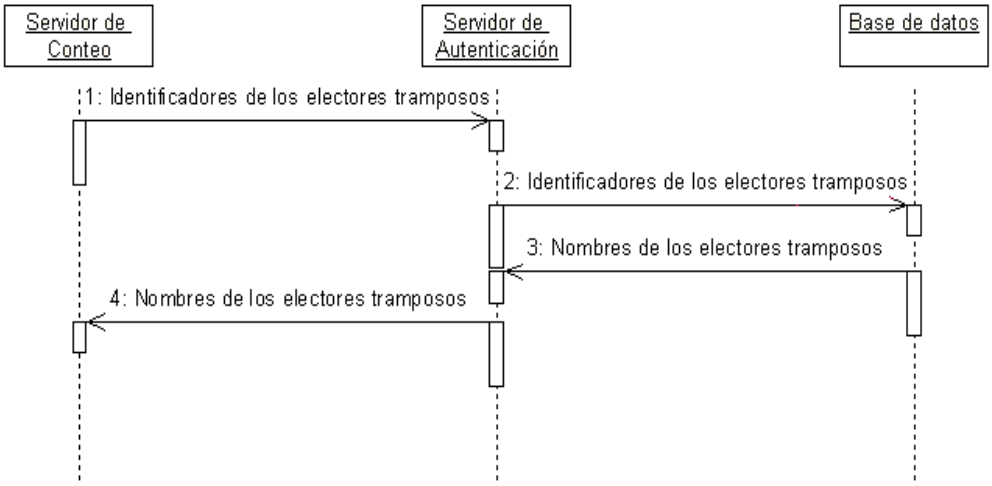

Figura 5.12: Diagrama de secuencia entre el SC y el SA

Por último se expone el diagrama de clases para el SC. En este digrama (ver figura 5.13) se muestran las clases implementadas para llevar a cabo la fase de conteo. A continuación se presenta una descripción de las clases que no han sido explicadas anteriormente:

- *GuardarBoletoPre*: Se encarga de almacenar los boletos (recibidos a través de un archivo) en la base de datos correspondiente.
- *ServletAnalizador*: Este servlet es el responsable de localizar los boletos idénticos y los sospechosos. Además de que obtiene los identificadores de los electores tramposos.
- *ServletEmisor*: Envía los identificadores obtenidos (k_2's al SA para recibir posteriormente los nombres de los electores que emitieron su voto más de una vez.
- *FuncionMD5*: Aplica la función *hash* MD5 a una determinada cadena. Esta clase se utiliza para obtener los compendios de los boletos recibidos.

5.3. Aplicación elector

La *aplicación elector* es la que se encarga de realizar todas las operaciones por parte del votante. Por lo cual es necesario que ésta se ejecute en la máquina que se está utilizando para votar.

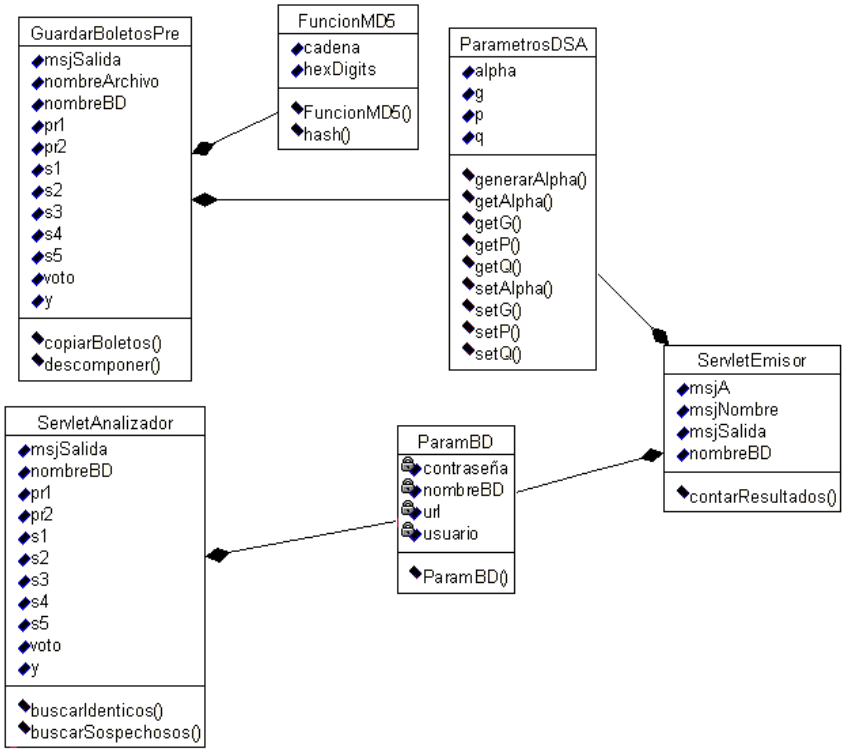

Figura 5.13: Diagrama de clases para el SC

Es por ello que, se diseñó e implementó un *applet* para computadoras personales y/o portátiles. Sin embargo, para asistentes personales digitales, debido a sus limitantes inherentes [21], se desarrolló una aplicación utilizando la clase *Frame*.

En la figura 5.14 se puede observar el diagrama de flujo para esta entidad. Como puede verse, el *elector* tiene que participar en dos fases para poder emitir su voto correctamente:

1. *Fase de autenticación*: Durante esta fase el elector envía su certificado digital y sus datos personales al SA. Verifica que no esté autenticado previamente para leer o generar sus parámetros. Después obtiene su llave privada y la llave pública del SA para poder generar su firma y los valores que le enviará a SA para que se los firme a ciegas. A continuación, envía su mensaje y espera la respuesta correspondiente. El mensaje de respuesta debe ser descompuesto para poder descifrar el valor k_2 (identificador del votante) que se ha asignado, y para descifrar las tres firmas a ciegas. Ahora, el *elector* procede a quitar el factor de opacidad a las firmas y a almacenar en un archivo sus parámetros.

2. *Fase de votación*: Al inicio de esta fase, el elector elige un voto y lo firma dos veces con diferentes llaves utilizando el esquema de firma digital DSA. Después genera

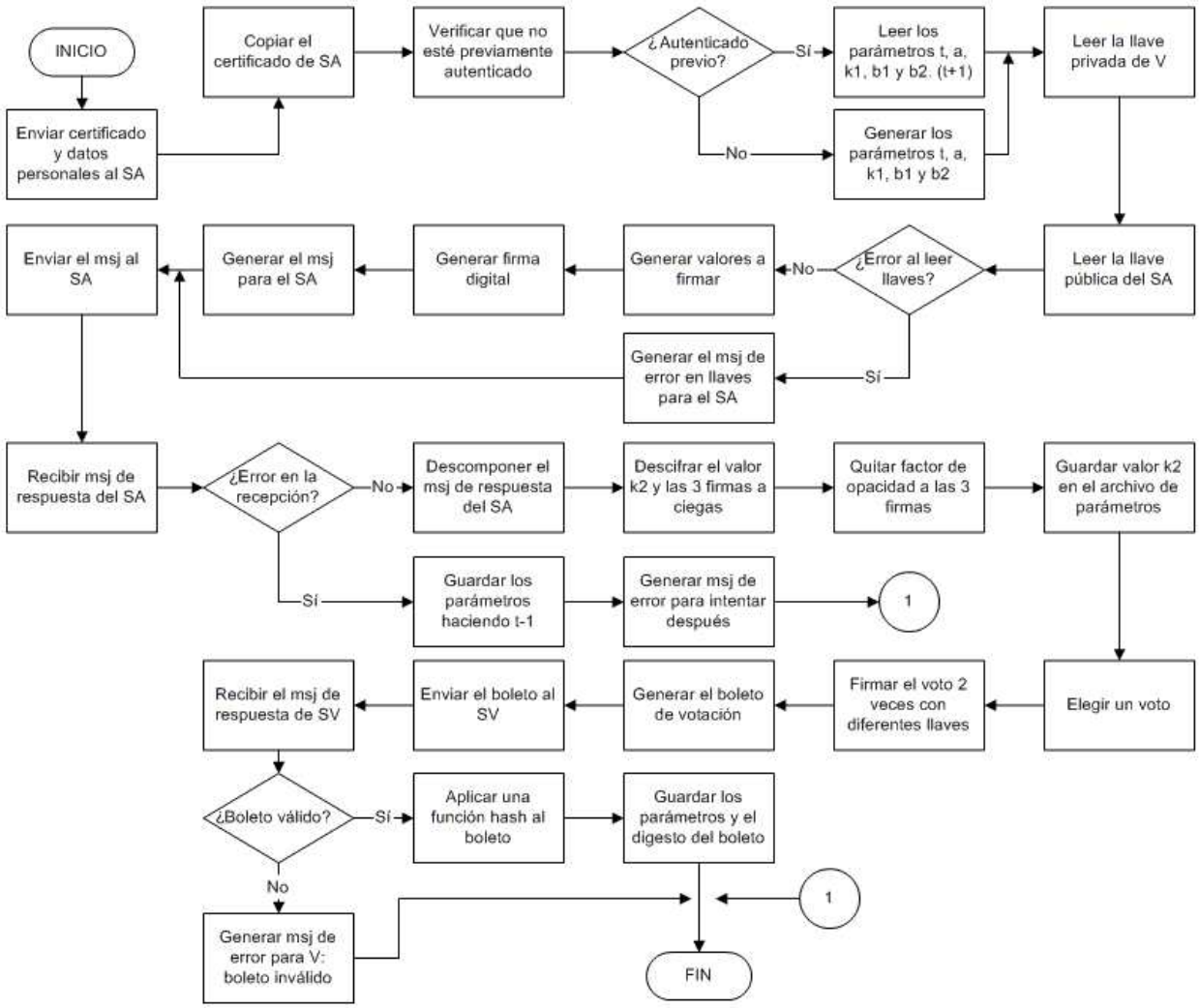

Figura 5.14: Diagrama de flujo de la *aplicación elector*

su boleto de votación y se lo envía al SV, el cual enviará su mensaje de respuesta posteriormente. Una vez recibida la respuesta de SV, el votante puede saber si su boleto ha sido válido o si alguna firma no ha verificado correctamente. Si el boleto es válido se le aplica una función *hash* y se guardan los parámetros junto con el compendio del boleto. Pero si SV indica que no todas las firmas del boleto han sido correctas, entonces se manda un mensaje de error al elector para darle a conocer la situación.

5.3.1. En computadoras con gran capacidad

Actualmente las computadoras personales y las portátiles cuentan con una capacidad casi ilimitada. Ya que constantemente aumenta su velocidad de procesamiento y la memoria en disco duro ha alcanzado hasta los 80 GB.

Por esto, para facilitar la participación del votante durante las elecciones haciendo uso de una PC o lap-top, se desarrolló un *applet* que realiza todas las operaciones requeridas.

Como se sabe, un *applet* es una aplicación que puede ser incluída en una página HTML y descargada para ejecutarse por un navegador web. Su código reside en un servidor, sin embargo, al momento de invocarlo, el código viaja a través de la red y se ejecuta en la máquina cliente. Estas características son ideales para nuestro sistema, ya que el elector debe seleccionar varios parámetros secretos para poder emitir su voto (véase sección 4.4).

Ahora bien, por motivos de seguridad para las personas que hacen uso de los *applets*, éstos están inhabilitados para:

- Ejecutar otro programa a partir de ellos mismos.
- Escribir en archivos I/O (input/output).
- Hacer llamadas a métodos nativos.
- Intentar abrir sockets a cualquier sistema diferente del que se ha obtenido el applet.

Por tal motivo SELES hace uso de *applets firmados* o *privilegiados*, ya que se requiere leer y escribir el archivo de parámetros que se encuentra en la máquina del elector.

Un applet firmado es aquél que contiene una firma digital, la cual puede ser aceptada por el usuario que ejecute el programa, y así conceder los privilegios necesarios para su correcta y completa ejecución. Antes de que un *applet firmado* sea descargado se muestra un mensaje solicitando la aceptación o el rechazo de su ejecución. En nuestro caso, el *applet* se firma con la llave privada del SA.

Otro aspecto importante de esta aplicación es la manera en que está construido. Por tal motivo se muestra en la figura 5.15 el diagrama de clases compone su estructura.

Las clases utilizadas para la *aplicación elector* son 5, sin embargo sólo 2 de ellas no han sido descritas previamente:

- *Marco*: Esta es la clase principal, por lo tanto tiene la función de generar y recibir los mensajes necesarios para poder llevar a cabo el proceso de votación adecuadamente.

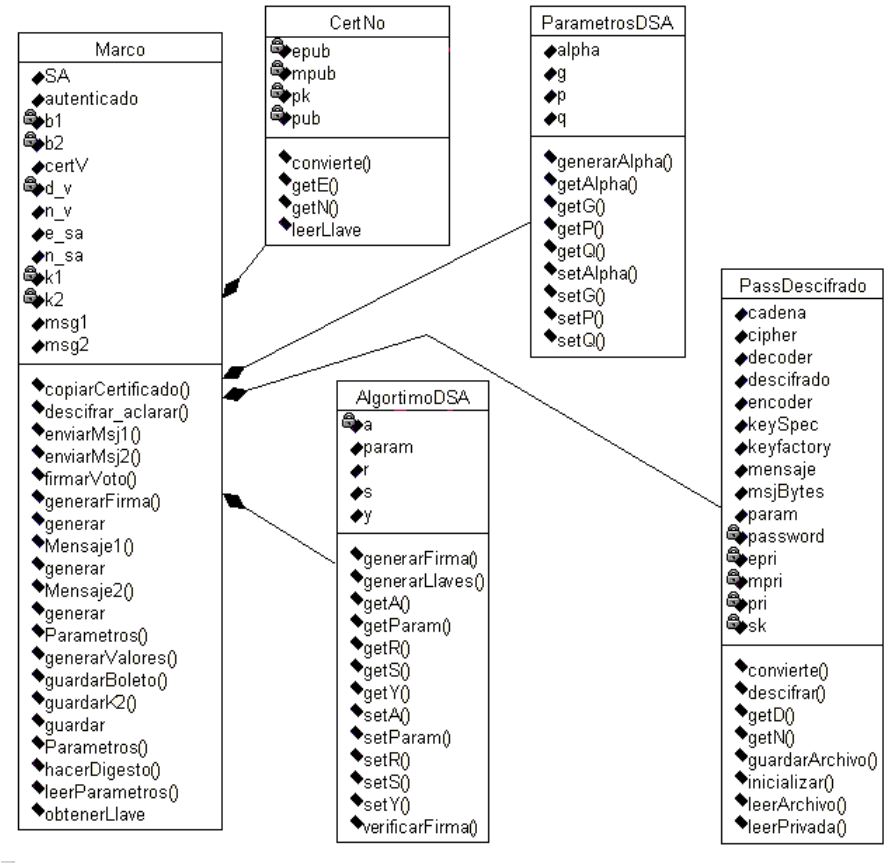

Figura 5.15: Diagrama de clases de la *aplicación elector* para computadoras de gran capacidad

- *PassDescifrado*: Se encarga de leer la llave privada cifrada por una contraseña (ver figura 2.2). Para descifrar la llave se necesita recibir la contraseña correcta, aplicarle una función *hash* (que en este caso es una función MD5), y utilizar el compendio como llave simétrica para así obtener la llave privada en claro.

5.3.2. En dispositivos móviles con capacidad limitada (PDA's)

SELES ha sido diseñado para que se puedan emitir votos desde asistentes personales digitales (PDA's). Por lo cual, se implementó una *aplicación elector* con características diferentes a las presentadas en la subsección anterior. Esto fue debido a que la mayoría de los navegadores de PDA's sólo soportan *applets* simples, pero no *applets firmados*.

Por lo anterior, en lugar de utilizar un *applet*, para PDA's se implementó una aplicación con la clase *Frame* para poder llevar a cabo todas las funciones del *elector*.

En la figura 5.16 se puede observar su diagrama de clases.

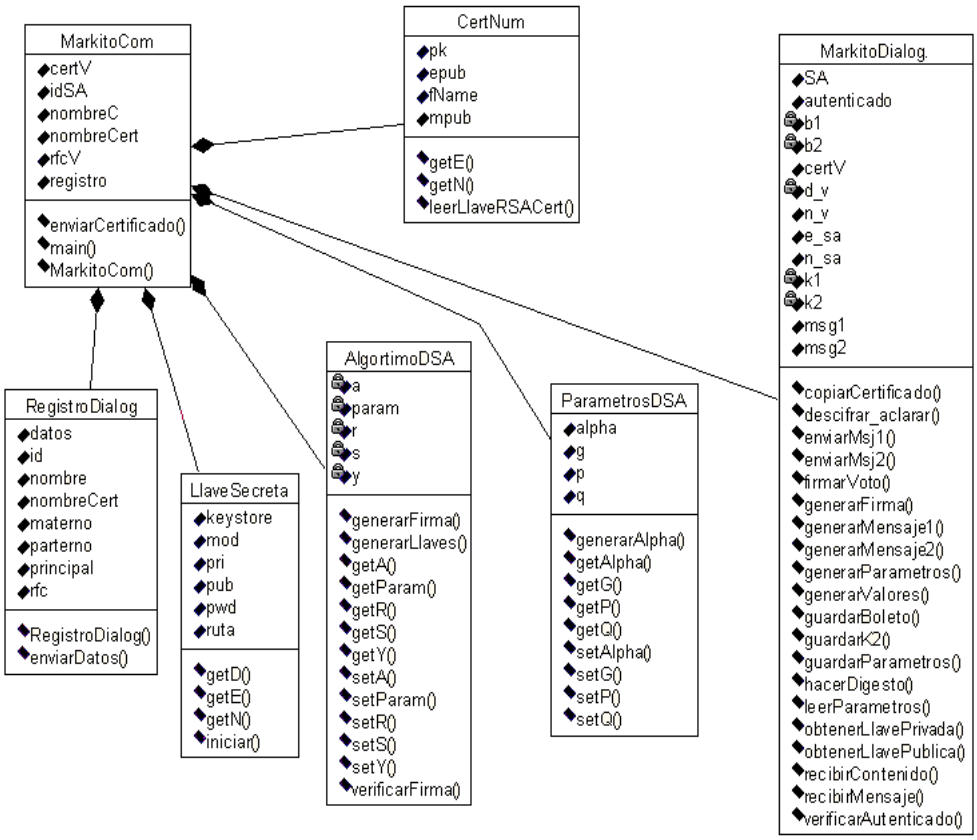

Figura 5.16: Diagrama de clases de la *aplicación elector* para PDA's

Las clases presentadas en la figura 5.16 que no se han mencionado con anterioridad son:

- *MarkitoCom*: Cuya función es enviar el certificado digital del votante al servidor de autenticación.
- *RegistroDialog*: Su responsabilidad es recopilar los datos personales del votante actual y enviarlos al SA para que sea registrado.
- *MarkitoDialog*: Se encarga de generar los parámetros y los mensajes necesarios para la comunicación con las autoridades SA y SV. Además, recibe y procesa los mensajes de respuesta que obtiene durante la fase de autenticación y votación.
- *CertNum*: Se utiliza para leer la llave pública contenida en los certificados digitales usados para PDA's.
- *LlaveSecreta*: Descifra la llave privada a través de una contraseña dada.

Esta *aplicación elector* no hace uso de las clases *CertNo* y *PassDescifrado* para leer llaves públicas y privadas respectivamente, debido a que para asistentes personales digitales se utilizan pares de llaves y certificados digitales generados por la herramienta *keytool*, y para computadoras de escritorio y portátiles se ocupan los generados por la autoridad certificadora ACERPAM. Para más detalles se puede consultar el apéndice B.

Por último se mencionará que, debido a que constantemente existen problemas de comunicación cuando se está utilizando Internet, 1) el manejo del archivo con los parámetros del votante, y 2) el uso de las estampas de tiempo, son técnicas fundamentales para proveer a SELES cierto grado de robustez.

5.4. Detalles de implementación

A continuación se presentan los detalles de implementación:

- Las tres *autoridades* del protocolo de comunicación fueron implementadas con *Servlets y JSP's* [8].
- El entorno de desarrollo para las *autoridades* y la *aplicación elector* para computadoras de gran capacidad fue la máquina virtual de Java con el JDK versión 1.5.0_01.
- Como servidor web se usó Apache Tomcat versión 5.5.
- Para el almacenamiento de la información se utilizó MySQL versión 1.4 como gestor de base de datos. Y MySQL Control Center 0.8.9-beta como interfaz gráfica para MySQL.
- Para la *aplicación elector* correspondiente a las PDA´s se hizo uso del entorno JeodeRuntime [36, 37] compatible completamente con PersonalJava[38, 39].
- Para electores que utilicen una PC o una lap-top, la aplicación que se requiere para poder emitir sus votos es un *applet* que se descarga después de haber sido registrado. Y para los que voten a través de un asistente personal digital, será necesario que previamente instalen en su dispositivo móvil la aplicación *Frame*, denominada *Markito*.
- Las PDA's que se utilizaron son: una iPAQ Pocket PC modelo h5550 y una SHARP modelo Zaurus SL-5500. Además, el código de la *aplicación elector* es igual para ambos dispositivos, salvo una pequeña diferencia en el uso de la clase *String* (cadena).
- La base de datos de SA se llama *EleccionesSA*, y se compone de las siguientes relaciones:

1. *sa_llaves*: Contiene los campos necesarios para especificar el exponente público, el módulo público y la ruta donde se encuentra el archivo con la llave privada cifrada.
2. *sa_padron*: Almacena la información personal de los votantes que pueden participar en determinadas elecciones.
3. *sa_datos*: Guarda los datos personales de los votantes que han enviado un mensaje al SA. Además de los valores contenidos en dicho mensaje.

- *EleccionesSV* es el nombre de la base de datos de SV, y contiene las relaciones:
  1. *sv_llavessa*: Almacena la llave pública del SA.
  2. *sv_datos*: Contiene los valores contenidos en los boletos de votación que se le enviaron al SV y que verificaron correctamente las 5 firmas.
- Finalmente, para el SC se tiene una base de datos denominada *EleccionesSC* y compuesta por las relaciones:
  1. *administradores*: Tiene registrados el nombre y el compendio de la clave de acceso de los posibles administradores encargados de enviar a SC los boletos válidos que ha recibido el SV.
  2. *sc_datos*: Almacena los valores contenidos en los boletos de votación, así como su compendio y algunas de sus características.
  3. *sc_tramposos*: Contiene los identificadores k_2 obtenidos a partir de los boletos sospechosos que se han identificado, así como los nombres de los electores que los enviaron en la fase de votación.

Capítulo 6

Análisis y evaluación final

Actualmente las elecciones electrónicas son viables para sustituir a las elecciones tradicionales. Por lo tanto, es importante que los sistemas para votar en línea posean las características propias de los esquemas convencionales. Además, se ha comprobado que mediante la emisión electrónica de votos se pueden alcanzar propiedades extras deseables, como la privacidad y seguridad física.

Son muchos los protocolos para elecciones electrónicas que se han propuesto, desafortunadamente pocos se han implementado. Lo anterior provoca cierta dificultad para comprobar las características de los esquemas propuestos, ya que es de todos sabido que el paso de propuestas a prototipos funcionales no siempre es inmediato. Sin embargo, algunos sistemas tales como Sensus[4] y REVS [13,14] pueden servir de referencia para comparar las características propias de SELES.

A pesar de lo anterior, SELES fue implementado para que, sin mayor esfuerzo, pueda ser utilizado en elecciones con un número mayor de participantes. En este trabajo se ha comprobado experimentalmente que SELES funciona adecuadamente para padrones de hasta 5,000 votantes.

En este capítulo se presenta en la sección 6.1 las pruebas realizadas a SELES. Después, se muestran los resultados alcanzados en dichas pruebas la sección 6.2. En la sección 6.3 se expone el correspondiente análisis de los resultados obtenidos y, finalmente se describe un caso de estudio en la sección 6.4.

6.1. Pruebas realizadas

Para poder realizar las pruebas a SELES se utilizaron los siguientes clientes:

1. Una computadora portátil Sony Notebook con procesador Pentium 4-M a 2 GHz, con S.O. Windows XP Profesional,

2. Una PDA iPAQ Pocket PC modelo h5550 con procesador Intel(R) PXA255 a 400 MHz, con S.O. Windows CE 4.2, y

3. Una PDA SHARP modelo Zaurus SL-5500 con procesador Strong ARM (SA-1110) a 206 MHz, con S.O. Linux 2.4 (Embedix).

Y para la fase de conteo de votos se utilizó como servidor de conteo (SC) la computadora portátil anteriormente mencionada.

Las pruebas que se realizaron consistieron en emitir votos desde los tres clientes para dos distintas elecciones simultáneas; en otras palabras, cada votante debía seleccionar dos candidatos para dos diferentes cargos en una sola sesión.

Para la fase de conteo se hicieron pruebas con 100, 200, 300, 400, 500, 600, 700, 800, 900, 1000, 1500, 2000, 3000, 4000, y 5000 boletos.

Además, se probó todo el sistema en sus diferentes etapas simulando y/o provocando fallos en la conexión a Internet, para así comprobar su robustez y tolerancia a fallos.

6.2. Resultados obtenidos

Originalmente, el protocolo de comunicación de SELES está compuesto de tres fases: la de autenticación, la de votación, y la de conteo y publicación. Sin embargo, en el sistema implementado, la primera fase se dividió en dos subfases: pre-autenticación y autenticación. La primera subfase consiste en el envío del certificado digital y los datos personales del votante al servidor de autenticación; y la segunda incluye el envío de la estampa de tiempo, los valores que SA firmará a ciegas, y la firma generada por el elector. Como puede notarse, en la primera subfase se envía la primera parte del mensaje indicado en el protocolo (ver sección 4.4), y en la segunda subfase es transferido el resto de la petición que hace el votante al SA.

En la tabla 6.1 se puede observar el tamaño de los mensajes que circulan por la red durante la fase de autenticación y la de votación.

Cuadro 6.1: Tamaño aproximado de mensajes para 1 votante

FASE	MENSAJE 1	MENSAJE 2
Autenticación	11.5Kb	4.5Kb
Votación	06.7Kb	—

Como se recordará, durante la fase de autenticación el elector envía una petición al SA (MENSAJE 1), y recibe de éste una de respuesta (MENSAJE 2); por lo cual son en total dos los mensajes que son transferidos en esta etapa.

En la fase de votación sólo se envía el boleto de votación (MENSAJE 1) al SV (véase figura 4.2).

En cuanto a las operaciones criptográficas que se realizan durante todo el proceso de votación, se realizó un análisis cuyos resultados se presentan en la tabla 6.2.

Cuadro 6.2: Operaciones criptográficas

FASE	VOTANTE	S. AUT.	S. VOT.
Autenticación	1 firma RSA 2 cifras RSA	1 verif. RSA 4 cifras RSA 3 fir. a ciegas	
Votación	4 descif. RSA 2 firmas DSA		3 verif. RSA 2 verif. DSA
TOTAL	9 operaciones	8 operaciones	5 operaciones

Como puede verse, entre el votante, el servidor de autenticación y el servidor de votación se realizan 22 operaciones criptográficas en total. De las cuales 11 se calculan a lo largo de la fase de autenticación, y las otras 11 durante la fase de votación.

Por último, en la figura 6.1 se muestran los tiempos que tarda SELES en validar y contar determinada cantidad de boletos durante la última fase del conteo final. Como puede observarse, SELES es capaz de contar correctamente 5,000 boletos con 2 votos cada uno en aproximadamente 2 minutos y 20 segundos. Este tiempo es considerablemente menor al que se tardaría un grupo de 5 personas en contar la misma cantidad de votos. Con esto se desea mostrar que efectivamente el conteo final que realiza SELES en elecciones electrónicas es, por mucho, más eficiente que el que se hace en elecciones con el esquema tradicional.

6.3. Análisis de los resultados

En esta sección se presentan un análisis de los resultados que se obtuvieron una vez realizadas las pruebas anteriormente mencionadas.

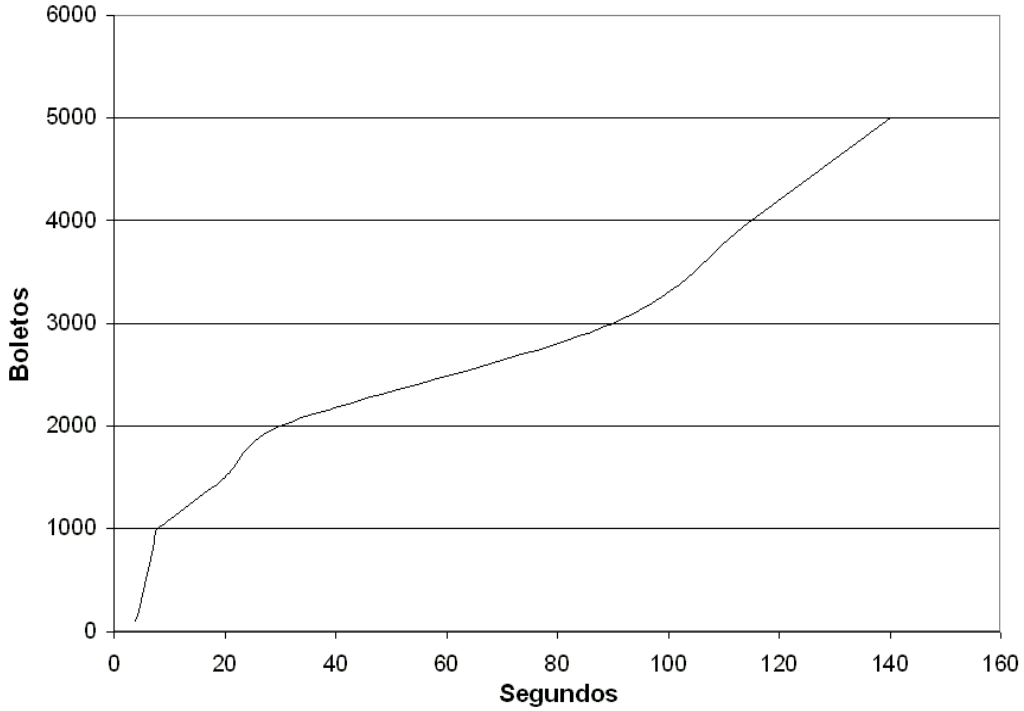

Figura 6.1: Tiempo requerido en la fase de conteo

6.3.1. Funcionalidad

Después de haber implementado el sistema, puede notarse que SELES posee las siguientes propiedades:

1. *Exactitud*: Dado que con las firmas a ciegas se asegura que el SV no aceptará ningún boleto que no contenga las 3 firmas realizadas por el SA, se garantiza que ningún boleto inválido será contado en los resultados finales, o bien, que algún boleto válido no será considerado en el conteo final. Además, se cuenta con un procedimiento para detectar 2 o más boletos válidos, pero emitidos por un mismo votante.

2. *Democracia*: Esta propiedad se logra ya que el SA verifica la identidad del votante, y así podrá comprobar que cumple con todos los requisitos solicitados para participar en las elecciones.

3. *Privacidad*: Al utilizar firmas a ciegas se garantiza que, dado un voto, no puede identificarse a la persona que lo emitió.

4. *Verificación*: Dado que el boleto que se genera contiene 5 firmas digitales, 3 con llaves elegidas sólo por el votante y 2 usando llaves generadas entre el votante y el SA, y el voto en claro (m); cada elector puede verificar que su voto fue contado de manera correcta al ser publicados los resultados. Ya que al finalizar las elecciones,

el SC no sólo publica los resultados finales, sino también todos los boletos válidos que recibió.

5. *Conveniencia*: Los votantes son capaces de terminar el proceso de votación en poco tiempo, en una sesión y con equipo mínimo.

6. *Flexibilidad*: Este sistema presenta la característica de flexibilidad debido a que no se basa en funciones homomórficas, sino en firmas a ciegas. Y con esto permite el uso varios formatos para la boleta de votación.

7. *Detección de 2 o más votos emitidos por un mismo votante*: El sistema posee esta propiedad ya que puede detectar si un votante ha emitido dos o más votos, y además es capaz de conocer la identidad del votante tramposo.

La siguiente tabla comparativa (6.3) muestra las propiedades que poseen algunos de los esquemas de votación más representativos. Los números de las propiedades corresponden a exactitud, democracia, privacidad, verificación, conveniencia, flexibilidad y detección de votos dobles (DT-VD) respectivamente.

Cuadro 6.3: Propiedades deseables

ESQUEMA	EXAC.	DEMC.	PRIV.	VERF.	CONV.	FLEX.	DT-VD
F. Homom.	√	√	√	√	√	×	×
Fujioka [6]	√	√	√	√	×	√	×
Sensus [4]	√	√	√	√	√	√	×
Karro [16]	√	√	√	√	√	√	×
REVS [13]	√	√	√	√	√	√	×
SELES	√	√	√	√	√	√	√

6.3.2. Robustez

Las pruebas que se realizaron para comprobar la robustez y la tolerancia a fallos de SELES arrojaron resultados satisfactorios. Ya que al introducir fallos en la conexión a Internet, durante las diferentes fases, se comprobó que el sistema no presentó vulnerabilidades que pudieran ser aprovechadas por usuarios maliciosos para alterar su funcionamiento correcto.

- *Fase de autenticación*: No se presentaron errores en la ejecución de esta fase debido a la creación y manejo del archivo con los parámetros de cada elector. Puesto que si un votante ya ha recibido valores firmados a ciegas por parte del SA, esta acción queda registrada en la base de datos correspondiente con el propósito de que, si el mismo votante vuelve a solicitar otras firmas a ciegas, SA le envíe de regreso los mismos

valores que recibió la primera vez. Además, si el votante genera datos diferentes, entonces SA puede darse cuenta y rechazar dicha petición, ya que la primera vez almacenó también dichos valores.

- *Fase de votación*: En esta fase SELES no presenta tanta vulnerabilidad, puesto que si existe un fallo en la comunicación con el SV, el elector puede mandar más tarde su boleto sin ningún problema. Este hecho no afecta de ninguna manera el funcionamiento del SV, ya que a la hora de enviarle los boletos válidos a SC, este último detectará los boletos idénticos para considerarlos una sola vez en la cuenta final.

- *Fase de conteo*: Si se presentan errores de comunicación entre el SV y el SC durante la fase de conteo, y es necesario enviar dos o más veces los boletos válidos al SC no se presentará ningún error, pues los boletos que se registren más de una vez en la base de datos tan sólo serán marcados como idénticos. Y como se recordará, el protocolo (sección 4.4) asegura que los boletos idénticos no podrán ser invalidados, tan sólo serán contados una sola vez. Por lo tanto, los resultados finales serán los mismos.

6.3.3. Comparación con otros esquemas

A continuación se presenta la tabla comparativa 6.4, en la cual se puede observar el número total de pares de llaves, contraseñas o claves, y autoridades que requieren los protocolos Sensus, REVS y SELES. También se compara el número de veces que se envía el voto a través de la red durante todo el proceso.

Cuadro 6.4: Tabla comparativa

ESQUEMA	PAR LLAV.	CONTRSÑ.	AUTORID.	TRANSMS.
Sensus [4]	$1+N$	1	3	$6N$
REVS [13]	$1+(t+1)N$	i	$4t$	$(2+2i)N$
SELES	$1+2N$	1	3	N

En la tabla 6.4 debe considerarse a N como el número de elecciones que se estén llevando a cabo. Por ejemplo SELES requiere que, para una elección ($N = 1$), el elector posea un par de llaves propio y genere dos pares más para firmar su voto dos veces durante la fase de votación. Si se llevaran a cabo tres elecciones al mismo tiempo (con diferente planilla), entonces el elector necesitaría, aparte del par de llaves que ha obtenido de la autoridad certificadora, generar 6 pares de llaves para firmar su voto dos veces por cada elección.

El valor de i, para el caso de REVS, corresponde a un valor mayor a $\frac{t}{2}$, donde t es el número total de entidades de cada tipo (*Ballot Distributor*, *Administrators*, *Anonymizers*

y *Counters*) que participarán en una elección. Cabe mencionar que el valor de t se elige dependiendo del grado de robustez que se requiera [13, 14].

Es claro ver que Sensus y SELES utilizan un menor número de pares de llaves por elección.

En cuanto al número de contraseñas requeridas por un votante, REVS se torna un poco ineficiente ya que necesita i claves, es decir, una por cada entidad *Administrador*, mientras que Sensus y SELES sólo usan una para descifrar la llave privada del elector.

El número de autoridades que contiene REVS es mayor en comparación con el de Sensus y el de SELES, pues debido a su alto grado de robustez y tolerancia a fallos necesita un número considerable de entidades reguladoras del proceso de votación. Sin embargo con el incremento en el número de participantes en una determinada elección, este hecho puede tornar ineficiente su desempeño en comparación con los otros dos sistemas.

Por último, en relación con la transmisión del voto a través de la red, se puede ver que únicamente SELES lo hace sólo una vez por cada elección que se lleve a cabo. Para el caso de REVS con $i = 2$ y $t = 3$, la transmisión del voto es igual a la de Sensus; sin embargo, con valores mayores para i y t, el número de veces que se transfiere el voto aumenta considerablemente.

La tabla 6.5 muestra los resultados correspondientes en cuanto al tráfico total sobre la red durante el proceso de votación.

Cuadro 6.5: Tamaño de mensajes transferidos

VOTANTES	FASE AUT.	FASE VOT.	TOTAL
1	2.00 KB	0.84 KB	2.84 KB
100	0.19 MB	0.08 MB	0.27 MB
1000	1.95 MB	0.82 MB	2.77 MB
2000	3.90 MB	1.64 MB	5.54 MB
3000	5.86 MB	2.45 MB	8.31 MB
4000	7.81 MB	3.27 MB	11.08 MB
5000	9.76 MB	4.09 MB	13.85 MB

6.4. Caso de estudio

Elecciones electrónicas en la Sec. de Computación

En esta sección se presenta un caso de estudio realizado con el objetivo de probar el buen funcionamiento y la eficiencia de SELES.

SELES fue adaptado para llevar a cabo un proceso de votación entre los estudiantes de la Sección de Computación del Departamento de Ingeniería Eléctrica, perteneciente a la Unidad Zacatenco del Centro de Investigación y Estudios Avanzados (Cinvestav).

6.4.1. Descripción

Como caso de estudio, en la Sección de Computación se convocó a estudiantes del plan de Maestría y Doctorado a participar en unas elecciones locales para elegir a una persona encargada de representar a cada grupo ante el Coordinador Académico.

Para poder votar, cada estudiante tuvo dos días para generar y descargar su par de llaves y su certificado digital a través de la autoridad certificadora **ACERPAM**.

Después, se establecieron tres días para que los participantes pudieran emitir su voto.

Pasados los tres días de votaciones, se publicaron los resultados en una página web accesible para todo el público en general.

6.4.2. Resultados

Durante estas elecciones se lograron detectar ciertos problemas de implementación y de seguridad que no se habían contemplado al inicio del diseño de este trabajo de tesis (véase apéndice D).

Uno de los más importantes fue el hecho de que si la versión de Java (del navegador del elector) no es la requerida por SELES, es imposible que los applets firmados se descarguen. Y si los applets no se visualizan, entonces el participante no puede emitir su voto.

Además, se identificó que las bases de datos, tanto de ACERPAM como de SELES, necesitan mayor seguridad. Pues en el caso de la base de datos de ACERPAM, ninguno de sus datos almacenados está cifrado. Y cabe la posibilidad de que algún intruso pueda acceder a la información. Por lo tanto, se consideró la posibilidad de agregar permisos y cortafuegos para proteger las bases de datos de ambos sistemas.

También se suscitó un problema al momento en que los electores proporcionaban sus datos, ya que en algunos casos, estos datos no coincidían con los registrados en la base de datos. Esto fue provocado porque los votantes agregaban u omitían acentos y/o segundos nombres; y entonces SELES no les permite votar. Para solucionar esto se necesitaría validar los datos personales de los usuarios y, agregar un formato estándar de texto a la base de datos. Adicionalmente, se podrían ligar los dos sistemas (ACERPAM y SELES), si éstos compartieran la misma tabla en la que se encuentra almacenada la información de los electores. Con esto último, también se evitaría que un cliente de la autoridad certificadora tuviera la oportunidad de generar más de un certificado, pues al registrarse en ACERPAM se le agregaría a su registro una marca de que ya ha generado su certificado digital.

Otro problema que se detectó, fue que en ambos sistemas las diferentes URL's están visibles al usuario. Hecho que podrían aprovechar usuarios maliciosos para provocar fallos o irregularidades dentro de las aplicaciones. Por lo cual, se consideró hacer uso de sesiones; para así controlar el tiempo en el que un usuario permanece activo dentro del sistema.

Sin embargo, al final de este proceso de votación se considera que la funcionalidad de SELES fue exitosa, puesto que:

- Se obtuvo un conteo final de votos exacto. Es decir, no se contaron votos repetidos ni votos enviados por un mismo elector.
- Se logró que sólo los estudiantes activos pudieran emitir sus respectivos votos.
- No se violó la privacidad de ningún participante.
- La fase de verificación se llevó a cabo sin ningún problema.
- La flexibilidad en el formato del voto estuvo presente.
- Se logró identificar al estudiante que voto dos veces, y eliminar sus votos de la cuenta final.

Capítulo 7

Conclusiones

El presente trabajo de tesis consiste en el diseño y la implementación de un Sistema para ELecciones Electrónicas Seguras (SELES) a mediana escala, mediante el cual se pueden realizar votaciones a través de Internet mediante computadoras de escritorio, computadoras portátiles y asistentes personales digitales (PDA's).

El objetivo principal de SELES es poseer las mejores características de los sistemas de votación tradicionales y, adicionalmente, ofrecer algunas otras que no siempre son propias de los sistemas convencionales, pero que sin embargo son deseables para poder llevar a cabo elecciones, tales como la privacidad y seguridad físicas.

SELES hace uso de llaves asimétricas RSA de 1024 bits y certificados digitales para realizar la autenticación; utiliza firmas a ciegas con RSA y el esquema de firma digital DSA para ofrecer privacidad al participante y detectar los votos que han sido emitidos por un mismo elector; y el criptosistema DES en conjunto con la función resumen MD5 para realizar el cifrado/descifrado de la llave privada del votante.

Además, SELES es eficiente en su fase de conteo, ya que es capaz de contar correctamente 5,000 votos en aproximadamente 140 segundos. Por lo que, comparado con los esquemas tradicionales, resulta ser una mejor alternativa.

Las contribuciones del presente trabajo de tesis pueden sumarizarse de la siguiente manera:

- Proveer un prototipo funcional para llevar a cabo elecciones electrónicas a través de Internet de forma conveniente.
- Aumentar la participación de los electores más ocupados en votaciones, debido a que SELES permite emitir votos desde PDA's.
- Implementación de un conjunto de clases en lenguaje Java para dispositivos móviles ligeros (PDA's), encargadas de realizar algunas operaciones criptográficas y manipular los datos de certificados digitales con el estándar X.509 v3.

- Desarrollo de clases y métodos de alto nivel, en lenguaje Java, para realizar operaciones criptográficas más flexibles.

Trabajo Futuro

Los siguientes puntos se pueden considerar para realizar mejoras a este trabajo de tesis:

- Dado que en el protocolo de votación implementado hace uso de dos criptosistemas de llave pública, es decir, 1) RSA para la autenticación y la generación de firmas a ciegas, y 2) DSA para firmar el voto y detectar los votos emitidos por un mismo elector; sería recomendable sólo utilizar uno, DSA. Para lo cual, sólo se necesitaría implementar un algoritmo para la generación de firmas a ciegas con DSA.

- En caso de que se lograra sólo utilizar DSA en el protocolo de votación, entonces, podría pensarse en sustituír este criptosistema por criptografía de curvas elípticas (CCE), la cual ha demostrado ser una mejor alternativa comparada con RSA e incluso con DSA.

- Debido a que los certificados utilizados se diseñaron de acuerdo a las necesidades del proyecto, haciendo uso del estándar X.509 v3, se podría sugerir la posibilidad de que éstos estuvieran apegados al estándar ASN. 1 para que pudieran ser legibles desde cualquier plataforma.

- Las pruebas realizadas a este proyecto incluyeron una conexión inalámbrica a Internet, sin embargo, para elecciones más pequeñas y exclusivas se podría analizar la posibilidad de que la comunicación entre los votantes y las autoridades fuera a través de una red Ad-Hoc.

- Se podría agregar a SELES la posibilidad de elegir los criptosistemas de llave pública, la función *hash*, y el algoritmo simétrico a utilizar durante un proceso de votación determinado.

- Se recomendaría agregar a la *aplicación elector* que se ejecuta sobre PDA's, la funcionalidad de poder hacer uso de los certificados digitales emitidos por cualquier autoridad certificadora que utilice el estándar X.509 v3.

- La escalabilidad de SELES podría ser aumentada en gran medida a través del aumento en el número de servidores de votos (SV's). Sin embargo, tendría que analizarse el comportamiento de las demás entidades participantes al efectuar este último cambio.

Apéndice A

Tecnología inalámbrica

Actualmente, el mercado de las comunicaciones inalámbricas ha tenido un crecimiento enorme. La tecnología inalámbrica virtualmente es capaz de alcanzar cualquier punto sobre la superficie de la Tierra. Cada día, cientos de millones de personas intercambian información por medio de PDA's, teléfonos celulares y otros productos para comunicación inalámbrica. Las redes inalámbricas (WLAN's *Wireless Local Area Network*) se han extendido rápidamente y ampliamente a pesar de la recesión en la economía de las telecomunicaciones en el mundo.

En sus inicios, las aplicaciones de las redes inalámbricas fueron confinadas a industrias y grandes almacenes. Hoy en día, las redes WLANs son instaladas en universidades, oficinas, hogares y hasta en espacios públicos. Las WLAN's típicamente consisten de computadoras portátiles o de escritorio que se conectan a dispositivos fijos llamados "puntos de acceso"(*access points*) vía señales de radio o infrarrojo. Las implementaciones de las WLAN's abarca todas las modalidades posibles desde las PAN's (*Personal Area Networks*), MAN's (*Metropolitan Area Network*)... hasta las WAN's (*Wide Area Networks*). Las PAN's, también conocidas como piconets, son pequeñas redes inalámbricas de corto alcance generalmente utilizadas para uso en interiores a pocos metros; un ejemplo de PAN podría ser un teléfono celular y una PDA conectados a través de tecnología Bluetooth. Mientras que las redes inalámbricas tipo WAN y MAN consisten de torres y antenas que transmiten ondas de radio o usan tecnología de microondas para conectar redes de área local, utilizando enlaces punto-punto y punto-multipunto [41].

Expertos en el campo siguen haciendo énfasis en los problemas inherentes de las tecnologías inalámbricas, tales como las limitaciones de ancho de banda disponible, problemas con interferencia y seguridad de la información transmitida, etc. Sin embargo, muchas de esas barreras que han inhibido el crecimiento de la tecnología inalámbrica están siendo resueltos. Se están superando las cuestiones que giraron alrededor de la estandarización y un número creciente de compañías están ofreciendo una variedad de soluciones de hardware y software.

Otro atractivo importante de los productos WLAN es la interoperatibilidad. Gracias al desarrollo de estándares, pueden mezclarse dispositivos inalámbricos de diversos fabricantes haciendo un acceso más directo y transparente con la tecnología.

A.1. Estándar IEEE 802.11

El protocolo IEEE 802.11 es un estándar de comunicaciones de la IEEE que define el uso de los dos niveles más bajos de la arquitectura OSI (capas física y de enlace de datos), especificando sus normas de funcionamiento en una WLAN. En general, los protocolos de la rama 802.x definen la tecnología de redes de área local.

Figura A.1: Configuración *ad-hoc*

La topología básica de una red 802.11 es la que se muestra en la figura A.1. Un conjunto de servicios básicos (BSS *Basic Service Set*), consiste en dos o más nodos inalámbricos, los cuales tienen que reconocerse unos a otros y establecer una comunicación punto a punto a través del medio inalámbrico. Este tipo de red es comúnmente llamada *ad-hoc*, o conjunto de servicios básicos independiente (IBSS *Independent Basic Service Set*).

Un conjunto de servicios básicos de infraestructura, mostrado en la figura A.2, es un BSS con un componente llamado "punto de acceso"(*access point*). El punto de acceso proporciona una función de reenvío local para el BSS. Todos los nodos en este tipo de red se comunican a través del punto de acceso y no directamente. De esta manera todos los mensajes, intercambiados entre las estaciones, son reenviados por el punto de acceso. Con esta función de reenvío local se logra doblar el rango de alcance del IBSS.

El estándar original de este protocolo data de 1997, tenía velocidades de 1 hasta 2 Mbps y trabajaba en la banda de frecuencia de 2.4 GHz. En la actualidad ya no se fabrican productos utilizando este estándar.

La siguiente modificación apareció en 1999 y es designada como IEEE 802.11b, esta especificación tiene velocidades de 5 hasta 11 Mbps y trabaja en la frecuencia de 2.4 GHz.

Figura A.2: Configuración de infraestructura

También se realizó la especificación 802.11a sobre una frecuencia de 5 GHz que alcanza los 54 Mbps; sin embargo resultaba incompatible con los productos de la especificación 802.11b y por motivos técnicos casi no se desarrollaron productos. Posteriormente se incorporó un estándar a esa velocidad y compatible con la 802.11b que recibiría el nombre de 802.11g. En la actualidad la mayoría de productos son de la especificación 802.11b y 802.11g.

En nuestros días, la extensión 802.11i es el nuevo estándar del IEEE. Incluye el nuevo algoritmo de cifrado AES (Advanced Encryption Standard). Este aspecto es importante puesto que significa que dispositivos con restricciones de cómputo no podrán utilizarlo. Para asegurar la integridad y autenticidad de los mensajes, la norma utiliza Counter-Mode with Cipher Block Chaining Message Authentication Code Protocol (CCMP). Además, el 802.11i incluye soporte para los conjuntos de servicios básicos independiente y de infraestructura.

Por último, actualmente se está desarrollando la 802.11n, que se espera que alcance los 500 Mbps.

A.2. Seguridad

Uno de los problemas las WLAN's es precisamente la seguridad ya que cualquier persona con una terminal inalámbrica podría comunicarse con un punto de acceso privado si no se disponen de las medidas de seguridad adecuadas.

Para poder considerar una red inalámbrica como segura, se debería cumplir con los siguientes requisitos:

- Las ondas de radio deben confinarse tanto como sea posible. Esto es difícil de lograr totalmente, pero se puede lograr una solución razonable empleando antenas direccionales y configurando adecuadamente la potencia de transmisión de los puntos de acceso.

- Debe existir algún mecanismo de autenticación en doble vía, que permita al cliente verificar que se está conectando a la red correcta, y a la red constatar que el cliente está autorizado para acceder a ella.

- Los datos deben viajar cifrados por el aire, para evitar que equipos ajenos a la red puedan capturar datos mediante escucha pasiva.

Existen varios métodos para lograr la configuración segura de una red inalámbrica [42]; cada método logra un nivel diferente de seguridad y presenta ciertas ventajas y desventajas. Algunos de ellos son:

1. *Filtrado de direcciones MAC*: Este método consiste en la creación de una tabla de datos en cada uno de los puntos de acceso a la red inalámbrica. Dicha tabla contiene las direcciones MAC (*Media Access Control*) de las tarjetas de red inalámbricas que se pueden conectar al punto de acceso.

2. *Wired Equivalent Privacy (WEP)*: El algoritmo WEP forma parte de la especificación 802.11, y se diseñó con el fin de proteger los datos que se transmiten en una conexión inalámbrica mediante cifrado. WEP opera en la segunda capa del modelo OSI y es soportado por la gran mayoría de fabricantes de soluciones inalámbricas.

   El algoritmo WEP, aparentemente, resuelve el problema del cifrado de datos entre emisor y receptor. Sin embargo, existen situaciones que hacen que WEP no sea seguro en la manera que es empleado en la mayoría de aplicaciones. Existen en este momento diversas herramientas gratuitas para romper la llave secreta de enlaces protegidos con WEP. Y en la actualidad se considera que este protocolo ha sido roto.

3. ***Las VPN***: Una red privada virtual (Virtual Private Network, VPN) emplea tecnologías de cifrado para crear un canal virtual privado sobre una red de uso público. Las VPN resultan especialmente atractivas para proteger redes inalámbricas, debido a que funcionan sobre cualquier tipo de hardware inalámbrico y superan las limitaciones de WEP.

4. *802.1x*: Protocolo de control de acceso y autenticación basado en la arquitectura cliente/servidor, que restringe la conexión de equipos no autorizados a una red. El protocolo fue inicialmente creado por la IEEE para uso en redes de área local alambradas, pero se ha extendido también a las redes inalámbricas. Muchos de los puntos de acceso que se fabrican en la actualidad ya son compatibles con 802.1x.

5. *WPA (WI-FI Protected Access)*: WPA es un estándar propuesto por los miembros de la Wi-Fi Alliance (que reúne a los grandes fabricantes de dispositivos para WLAN) en colaboración con la IEEE. Este estándar busca subsanar los problemas de WEP, mejorando el cifrado de los datos y ofreciendo un mecanismo de autenticación.

Para solucionar el problema de cifrado de los datos, WPA propone un nuevo protocolo para cifrado, conocido como TKIP (*Temporary Key Integrity Protocol*). Este protocolo se encarga de cambiar la clave compartida entre punto de acceso y cliente cada cierto tiempo, para evitar ataques que permitan revelar la clave. Igualmente se mejoraron los algoritmos de cifrado de trama y de generación de los IVs, con respecto a WEP.

A.3. Dispositivos móviles ligeros

Los constantes avances tecnológicos han permitido la creación de una gran variedad de dispositivos móviles equipados con tecnología inalámbrica. Como ejemplos tenemos: computadoras portátiles, asistentes personales digitales (PDA's), teléfonos celulares inteligentes y otros nuevos dispositivos que aparecen día con día.

La popularidad de estos dispositivos y su convergencia con tecnologías de telecomunicaciones han comenzado a desplazar a las computadoras fijas como el principal medio para el procesamiento y la transmisión de información digital. Dando lugar a nuevos escenarios los cuales se caracterizan por que ahora el usuario se puede trasladar junto con los servicios computacionales que le son útiles para el desarrollo de alguna tarea que antes implicaba el tener una infraestructura de comunicaciones y computacional preestablecida. Además, estas nuevas características tienen un gran impacto en las aplicaciones, en el desarrollo de los sistemas, etc.

Actualmente, los dispositivos móviles pueden clasificarse en: pesados y ligeros. Un ejemplo de dispositivo móvil pesado es una computadora portátil (lap-top), ya que no es tan portable como un teléfono celular y además su capacidad de procesamiento es igual o, en algunas ocasiones, superior a la de una computadora de escritorio.

Por otro lado, entre los dispositivos móviles ligeros están:

- Tarjetas inteligentes.
- Teléfonos celulares inteligentes.
- Pagers.
- Agendas electrónicas.
- Asistentes personales digitales (PDA's).

A.4. Perspectivas

La tecnología inalámbrica está disponible hoy en día y es sólo el principio de una tendencia creciente. Diversos estándares prometen un gran ancho de banda para permitir un

sinfín de nuevas aplicaciones. Aunque todavía existen varios obstáculos que hay que vencer como la seguridad e interferencia, las WLANs ofrecen por lo pronto una comunicación eficiente tanto en interiores como exteriores. Los precios de los productos WLAN se han estado reduciendo enormemente, y estos precios continuarán bajando conforme se alcance el consumo masivo del software y hardware basados en tecnologías inalámbricas. Cuando se evalúa una solución inalámbrica que satisfaga nuestras necesidades de comunicación, es muy importante tener en cuenta los estándares y tecnologías de más penetración. Esta sabia decisión ahorrará dinero, tiempo y problemas de incompatibilidad y nos brindará comunicación rápida, eficiente y transparente.

Apéndice B

Llaves y certificados utilizados

Para poder llevar a cabo el proceso de votación se requiere, además de los tres servidores propios del protocolo implementado, de una autoridad certificadora. Esta última es para proveer las llaves y los certificados digitales de los votantes que participarán en determinadas elecciones.

En la fase de autenticación, es decir, durante la interacción entre el votante y el servidor de autenticación SA, se requiere el uso del certificado digital del elector para comprobar que la firma que envió fue hecha con su llave privada correspondiente a la llave pública que está contenida en su certificado digital. Si la firma es correcta, entonces SA le puede regresar al votante los valores que recibió pero ya adecuadamente firmados a ciegas.

Sin embargo debido a que SELES permite realizar la emisión de un voto a través de una computadora de gran capacidad y/o de un asistente personal digital (PDA), fue necesario hacer uso de dos autoridades certificadoras. Puesto que, a causa de las limitantes propias de los asistentes personales digitales, no fue posible que estos últimos utilizarán las llaves y los certificados digitales generados por la autoridad certificadora ACERPAM.

B.1. ACERPAM

ACERPAM es una autoridad certificadora implementada con Java haciendo uso de JSP's y Servlets [40]. Sus funciones son:

1. Generar certificados digitales X.509 para llaves públicas proporcionadas por los clientes.

2. Generar pares de llaves y sus respectivos certificados digitales X.509.

3. Renovación de certificados.

4. Revocación de certificados.

5. Administración de la lista de revocación.

Figura B.1: Página principal de ACERPAM

Ahora bien, para que a un cierto cliente se le pueda generar un par de llaves y el certificado digital, éste debe solicitarlo desde la página principal de ACERPAM (ver figura B.1).

Una vez que ha elegido la opción correcta, se le solicitarán sus datos personales (ver figura B.2), y una contraseña para cifrar la llave privada recién generada (ver figura B.3). Esta última acción se hace mediante un *applet* firmado, ya que se requiere copiar a un directorio local el certificado de AC con el propósito de cifrar la contraseña con la llave pública de AC y que, de esta manera, viaje a través de la red de manera segura.

El proceso de cifrado de la llave privada se realiza aplicando a la contraseña una función hash MD5, y tomando el compendio resultante como llave simétrica para cifrar dicha llave.

Una vez que se ha enviado la contraseña, y no ha ocurrido ningún error, el cliente puede descargar el par de llaves y el certificado digital que se le han generado (ver figura B.4).

Figura B.2: Solicitud de datos personales

B.2. KEYTOOL

Los certificados usados por los electores que voten a través de un asistente personal digital son los generados por la herramienta *keytool* incluída desde el JDK 1.2, la cual reemplaza completamente a la herramienta *javakey* incluída en el JDK 1.1 (ver figura B.5).

Lo anterior es debido a que las PDA's, a causa de las restricciones que presentan, no pueden leer las llaves privadas ni los certificados generados por ACERPAM, ya que las librerías necesarias no están incluídas en el paquete *PersonalJava*.

Para generar una llave pública y una llave privada a través de la herramienta *keytool*, se hace uso del siguiente comando:

```
keytool -genkey {-alias alias} {-keyalg keyalg} {-keysize
    keysize} [-dname dname] [-keypass keypass] {-keystore
    keystore} [-storepass storepass]
```

donde,

- `alias`: es el alias que se asignará al par de llaves que está a punto de ser creado.

Figura B.3: Solicitud de la contraseña

- `keyalg`: indica el algoritmo que será usado para generar el par de llaves.
- `keysize`: especifica el tamaño de las llaves.
- `dname`: son los atributos para el certificado digital.
- `keypass`: es la contraseña para cifrar la llave privada.
- `keystore`: corresponde al nombre del archivo en donde se almacenará el par de llaves.
- `storepass`: especifica la contraseña para acceder al archivo especificado por keystore.

Para exportar el certificado se requiere la siguiente instrucción:

```
keytool -export {-alias alias} {-file cert_file} {-keystore
    keystore} [-storepass storepass]
```

donde,

- `cert_file`: es el nombre que se le asignará al certificado digital.

Figura B.4: Descarga de llaves y certificado

Figura B.5: Herramienta *keytool*

Apéndice C

SELES

Presentación

SELES es un Sistema para Elecciones Eléctronicas Seguras. El cual implementa protocolos de seguridad para brindar las propiedades de exactitud, democracia, privacidad, verificación, conveniencia, flexibilidad y detección de votos dobles emitidos por un sólo elector.

Está diseñado para soportar elecciones a mediana escala, aproximadamente 1000 participantes. Sin embargo puede extenderse hasta 10,000 votantes. La emisión de votos se hace en línea a través de Internet; y la autenticación se hace mediante certificados digitales proveídos por la autoridad certificadora ACERPAM y la herramienta *keytool*.

Características

- *Intefaz gráfica*: Amigable al usuario.
- *Independencia de la plataforma*: Implementado en Java.
- *Funcionalidad*: Puede ser ejecutado sobre computadoras de escritorio y/o portátiles, y asistentes personales digitales (PDA's).
- *Escalable*: Al agregar servidores de votación, el número de participantes puede aumentar considerablemente.
- *Ajustable*: Puede ser ajustado a necesidades específicas del cliente.
- *Simplicidad*: No requiere instalación previa de ningún software por parte del votante para la versión funcional sobre computadoras de escritorio y/o portátiles.
- *Propiedades ofrecidas*: Exactitud, democracia, privacidad, verificación, conveniencia, flexibilidad y detección de votos dobles emitidos por un sólo votante.

- *Algoritmos Asimétricos*: RSA.
- *Algoritmos para Firma/Verificación Digital*: RSA y DSA.
- *Firmas a ciegas*: RSA.
- *Funciones Hash*: MD5.
- *Certificados digitales*: Generados con el estándar X.509v3.
- *Compatibilidad de los certificados digitales*: Compatibles con la aplicación PGP.

Requerimientos del sistema para autoridades

- Sistema operativo: Linux, Windows 98, Windows 2000 o Windows XP.
- JVM instalada en con JDK versión 1.5.0_01
- Conexión a internet.
- Configuración de la base de datos.
- Configuración del servidor web *Tomcat* y del DBMS *MySQL*.

Requerimientos del sistema para votantes

- Sistema operativo: Linux, Windows 98, Windows 2000 o Windows XP. Windows CE 4.2 y Linux 2.4 (Embedix).
- Navegadores para votantes: Internet Explorer 6 y Netscape 4 (con la versión de Java jdk1.5.0_01).
- Conexión a internet.
- Para asistentes personales digitales (PDA's):
  1. Tarjeta de red inalámbrica habilitada y configurada como *Infraestructura*
  2. Instalación previa de la aplicación *Markito*.

Apéndice D

Diagnóstico técnico

Elecciones electrónicas en la sección de computación

A continuación se presentan las lecciones aprendidas al utilizar SELES en el caso de estudio descrito en la sección 6.4.

1. ACERPAM: No existe una base de datos con la información de las personas (clientes) que pueden generar y descargar certificados.

   Posible solución: Generar dicha base de datos para validar qué personas pueden generar y descargar certificados con ACERPAM. Con ésto se evitaría que un cliente pudiera generar más de un certificado.

2. ACERPAM: La seguridad de la base de datos es nula, ya que los datos almacenados no están cifrados.

   Posible solución: Cifrar la información de la base de datos, o bien, agregarle servicios de seguridad (como permisos, firewalls, etc.).

3. SELES: La seguridad de la base de datos sólo es debido que los datos que se almacenan están cifrados. Sin embargo, los intrusos tienen la posibilidad de ver toda la base de datos.

   Posible solución: Agregar a la base de datos servicios de seguridad (como permisos, firewalls, etc.).

4. SELES: Algunos usuarios no conocían el IDENTIFICADOR de las elecciones.

Posible solución: Si se trata de una sola elección se puede omitir la petición de este parámetro al elector. Pero si se trata de varios procesos de votación, lo que se haría es sustituir la caja de texto por un *combo box* que despliegue sólo los IDENTIFICADORES válidos.

5. ACERPAM Y SELES: Algunos usuarios que se registraron en la autoridad certificadora, ingresaban esos mismos datos en SELES; sin embargo, estas son dos aplicaciones independientes.

   Posible solución: Ligar las dos aplicaciones. Esto se podría hacer si compartieran las dos bases de datos, específicamente la tabla en la que se encuentran los datos personales de las personas que pueden generar y descargar certificados, y de las que pueden votar (*padrón electoral*).

6. ACERPAM Y SELES: Vulnerabilidades al momento de cambiar de una página Web a otra. Esto fue debido a que las diferentes URL's invocadas son visibles al usuario.

   Posible solución: Agregar un módulo para implementar el protocolo HTTPS en las dos aplicaciones, aumentando así la seguridad durante los cambios entre las páginas Web.

7. ACERPAM Y SELES: Problemas con el comportamiento de los acentos.

   Posible solución: Agregar a las bases de datos el formato estándar para texto.

8. ACERPAM Y SELES: Para algunos usuarios fue imposible que descargar los applets firmados debido a que las versiones de Java no eran las mismas, o porque los navegadores utilizados no lo permitían.

   Posible solución: Especificar claramente que versión de Java se necesita para descargar los applets involucrados. O bien, dar la opción de que los usuarios descarguen la aplicación para votar, como se hace cuando se vota a través de una PDA. Sin embargo, se tendría que implementar la *aplicación elector* para computadoras de escritorio y portátiles, ya que para este trabajo de tesis sólo se implementó la versión para PDA's.

9. ACERPAM Y SELES: Los datos introducidos por parte del usuario no concordaban con los registrados en la base de datos.

Posible solución: Validar los datos que ingresa el usuario cambiándolos a mayúsculas y quitándoles los acentos.

Bibliografía

[1] David Chaum, "Blind signatures for untraceable payments," *Advances in Cryptology, CRYPTO'82,* (1982): 199–203.

[2] David Chaum, "Elections with unconditionally-secret ballots and disruption equivalent to breaking RSA," *Advances in Cryptology – EUROCRYPT '88,* **vol.**(330), (Lecture Notes in Computer Science. Springer – Verlag, Berlin, 1988): 177–182.

[3] Ronald Cramer, Matthew Franklin, Berry Schoenmakers, y Moti Yung, "Multi-authority secrect-ballot elections with linear work," *Advances in Cryptology, EUROCRYPT '96,* **vol.**(1070), (Lecture Notes in Computer Science. Springer–Verlag, 1996): 72–83.

[4] Lorrie F. Cranor y Ron K. Cytron, "Sensus: A security-conscious electronic polling system for the Internet," *Proceedings of the Hawaii International Conference on System Sciences,* **vol.**(3), (Wailea, Hawaii, 1997): 561–570.

[5] Brandon W. DuRette, "Multiple Administrators for Electronic Voting," Tesis de Licenciatura, *Massachussetts Institute of Technology,* (1999).

[6] Atsushi Fujioka, Tatsuaki Okamoto, y Kazuo Ohta, "A practical secret voting scheme for large scale elections," *Advances in Cryptology, AUSCRYPT '92,* **vol.**(718), (Lecture Notes in Computer Science. Springer – Verlag, Berlin, 1993): 244–251.

[7] Darrel Hankerson, Alfred Menezes y Scott Vanstone, *Guide to Elliptic Curve Cryptography.* Springer Professional Computing, 2004.

[8] Phil Hanna, *JSP Manual de referencia.* McGraw–Hill Osborne Media, España, 2002.

[9] Rich Helton y Johennie Helton, *Java Security Solutions.* Wiley Publishing, Inc. Indianapolis, Indiana, 2002.

[10] Mark A. Herschberg, "Secure electronic voting over the word wide web," Tesis de Maestría, *Massachussetts Institute of Technology,* (1997).

[11] Martin Hirt y Kazue Sako, "Efficient receipt-free voting based on homomorphic encryption," *Advances in Cryptology, EUROCRYPT '00,* **vol.**(1807), (Lecture Notes in Computer Science. Springer–Verlag, 2000): 539–556.

[12] Kenneth R. Iversen, "A cryptographic Scheme for Computerized General Elections," *Advances in Cryptology, CRYPTO'91,* **vol.**(576), (Lecture Notes in Computer Science. Springer–Verlag, Berlin, 1992): 405–419.

[13] Rui Joaquim, André Zúquete y Paulo Ferreira, "REVS - A Robust Electronic Voting System," *Proceedings of IADIS International Conference e-Society 2003*, (Lisbon, Portugal, 2003): 95–103.

[14] Rui Joaquim, Ricardo Lebre, André Zúquete y Paulo Ferreira, "Internet Voting: Improving Resistance to Malicious Servers in REVS," *IADIS International Conference Applied Computing,*(Lisbon, Portugal, Marzo 2004).

[15] Wen-Shenq Juang y Chin-Laung Lei, "A secure and practical electronic voting scheme for real enviroments," *IEICE Transactions on Fundamentals,* **vol.**(E80–A, no. 1), (1997): 64–71.

[16] Jared Karro y Jie Wang, "Towards a Practical, Secure, and Very Large Scale Online Election," *Proceedings of the 15th Annual Computer Security Applications Conference (ACSAC'99),* (Scottsdale, AZ. Diciembre, 1999): 161–169.

[17] Costas Lambrinoudakis, Vassilis Tsoumas, Maria Karyda y Spyros Ikonomopoulos, "SECURE e-VOTING, The Current Landscape,", in the book *Secure Electronic Voting: Trends and Perspectives, Capabilities and Limitations, D. Gritzalis (Ed.), Kluwer Academic Publishers* (Greece, 2002).

[18] Iuon-Chung Lin, Min-Shiang Hwang y Chin-Chen Chang, "Security enhancement for anonymous secure e-voting over a network," *Computer Standards & Interfaces,* **vol.**(25, no. 2), (2003): 131–139.

[19] Shu-Chen Lin, "A Study on Proxy E-Voting Schemes," Tesis de Maestría, *Chaoyang University of Technology department of information management,* (Mayo, 2004).

[20] Carlos E. López Peza, "Sistema de Seguridad para Intercambio de Datos en Dispositivos Móviles," Tesis de Maestría, *Sección de Computación del Departamento de Ingeniería Eléctrica del Centro de Investigación y de Estudios Avanzados del IPN,* (México D.F., Abril, 2005).

[21] Qusay H. Mahmoud, *Learning Wireless Java.* O'Reilly, Estados Unidos de América, 2002.

[22] Guillermo Martínez Silva, "Diseño e Implementación de una Autoridad Certificadora en Plataformas Móviles," Tesis de Maestría, *Escuela de Graduados de Ingeniería y Arquitectura del Instituto Tecnológico y de Estudios Superiores de Monterrey,* (México D.F., Abril, 2005).

[23] Alfred Menezes, P van Oorschot, y S. Vanstone, *Handbook of Applied Cryptography.* CRC Press, 1996. Págs: 33-34.

[24] Yi Mu y Vijay Varadharajan, "Anonymous secure e-voting over a network," *Proceedings of the 14th Annual Computer Security Applications Conference,* (IEEE Computer Society, 1998): 293–299.

[25] Indrajit Ray, Indrakshi Ray y Natarajan Narasimhamurthi, "An anonymous electronic voting protocol for voting over the internet," *Proceedings of the Third International Workshop on Advanced Issues of E–Commerce and Web–based Information Systems,* (San Juan, CA, 2001): 188–190.

[26] Laura I. Reyes Montiel, "Estudio, diseño y evaluación de protocolos de autenticación para redes inalámbricas," Tesis de Maestría, *Sección de Computación del Departamento de Ingeniería Eléctrica del Centro de Investigación y de Estudios Avanzados del IPN,* (México D.F., Enero, 2004).

[27] Kazue Sako and Joe Kilian, "Secure voting using partially compatible homomorphisms," *Advances in Cryptology, CRYPTO '94,* **vol.**(839), (Lecture Notes in Computer Science. Springer–Verlag, 1994): 411–424.

[28] Kazue Sako and Joe Kilian, "Receipt-free mix-type voting scheme," *Proceedings in EUROCRYPT '95,* **vol.**(921), (Lecture Notes in Computer Science. Springer–Verlag, 1995): 393–403.

[29] Herbert Schildt, *Java 2, Manual de referencia.* Cuarta edición. Osborne McGraw–Hill, Madrid, 2001.

[30] Klaus Schmeh, *Cryptography and Public Key Infraestructure on the Internet.* Wiley, USA, primera edición, 2001.

[31] B. Schneier, *Applied Cryptography. Protocols, Algorithms, and Source Code in C.* Wiley, USA, segunda edición, 1996.

[32] Berry Schoenmakers, "A simple publicly verifiable secret sharing scheme and its Application to the electronic voting," *Advances in Cryptology, CRYPTO'99,* **vol.**(1666), (Lecture Notes in Computer Science. Springer–Verlag, 1999): 148–164.

[33] Wade Trappe y Lawrence C. Washington, *Introduction to Cryptography with Coding Theory.* Prentice–Hall, Upper Saddle River, New Jersey, 2002.

[34] Electronic Voting Repository,

http://www.tcs.hut.fi/∼helger/crypto/link/protocols/voting.html

[35] Firma Digital, http://www.tuguialegal.com/firmadigital1.htm

[36] Insignia Jeode Runtime Environment for WinCE, http://www.cs.unc.edu/∼lindsey/7ds/notes/jeode/

[37] Java Programming on the Sharp Zaurus,
http://developers.sun.com/techtopics/mobility/personal/articles/ ztutorial/

[38] J2ME Documentation CDC,
http://java.sun.com/j2me/docs/index.html

[39] J2ME PersonalJava, http://java.sun.com/products/personaljava/

[40] Reporte técnico: Claudia Patricia García-Zamora, Mónica Rivera-de la Rosa, "ACERPAM". Disponible en: http://delta.cs.cinvestav.mx/∼francisco/Repository/ReporteAC.pdf

[41] Revista: "RED", junio del 2002. Editorial RED S.A de C.V. Edición On-Line en http://www.red.com.mx/

[42] Revista: "S & T Ingeniería de Sistemas e Ingeniería Telemática, Universidad ICESI". Páginas 13-28. Disponible en: http://www.icesi.edu.co/es/publicaciones/