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

DISEÑO E IMPLEMENTACIÓN DE UN SISTEMA DE VOTACIÓN ELECTRÓNICA MODULAR Y DUAL, VERIFICABLE POR EL VOTANTE

MEMORIA PARA OPTAR AL TÍTULO DE INGENIERO CIVIL EN COMPUTACIÓN

CAMILO JOSÉ GÓMEZ NÚÑEZ

PROFESOR GUÍA:
ALEJANDRO HEVIA ANGULO

MIEMBROS DE LA COMISIÓN:
ERIC TANTER
MAURICIO PALMA LIZANA

SANTIAGO DE CHILE
2015

Resumen

La votación electrónica poco a poco se ha ido masificando a lo largo del mundo. Lamentablemente su uso en procesos eleccionarios de varios países, no ha estado exento de variadas críticas y cuestionamientos. Estos cuestionamientos vienen dados producto de varios estudios realizados a dichos sistemas, los cuales evidencian claras brechas de seguridad que pudieron haber vulnerado propiedades como privacidad del voto, integridad del proceso, entre otros.

Para subsanar lo anterior, una posible solución es darle al votante responsabilidad a la hora de asegurar que el sistema esté funcionando bien. En particular, pedirle al votante realizar una verificación previa de que su voto electrónico realmente representa lo que quiere, y no otra opción.

Para llevar eso a cabo, se diseñó e implementó un sistema de votación electrónica que posee la novedad de incluir un módulo donde el votante tiene la responsabilidad de verificar que su voto electrónico es consistente con la opción seleccionada. Además de esto, el sistema genera una copia en papel del voto (como en una votación clásica), para dotar al sistema de un mecanismo que preserve los votos si es que existiera alguna sospecha de mal funcionamiento de éste. Por otro lado, el sistema diseñado pretende ser altamente modular, donde que cada una de las partes es específica en su funcionamiento, haciendo más fácil su auditoría, además de disminuir un posible ataque general al sistema. Los requisitos de hardware necesarios para la implementación realizada son reducidos, bajando así los costos asociados a montar el sistema, y el software utilizado es sencillo. Por último, el sistema utiliza dispositivos cercanos a los usuarios, para así mejorar la experiencia y usabilidad.

Las pruebas realizadas muestran que aun hay muchas mejoras que realizar en el aspecto de usabilidad, ya que algunos pasos (los más novedosos) no son bien comprendidos por los usuarios, dificultando el proceso de votación. Además de eso se evidencian algunas otras mejoras a realizar para mejorar la seguridad de los datos que se van procesando en el sistema.

A pesar de lo anterior, el futuro del proyecto se muestra prometedor. El diseño realizado se preocupa, tanto del usuario final (el principal actor dentro del sistema), como de que la confianza de éste hacia el sistema se vea fortalecido cada vez que el votante se involucra en el proceso.

A mis padres y mi hermano, quienes nunca dudaron de mi.

Agradecimientos

Primero que todo mis agradecimientos van hacia Jorge, mi padre, y Ana, mi madre, quiénes a lo largo de toda mi carrera universitaria, siempre estuvieron apoyándome, incluso en la distancia, y que sabiendo poco de la disciplina que estudio, siempre estuvieron preocupados de mis quehaceres y que disfrutara lo que estaba haciendo. Además de ellos, especiales agradecimientos a Pablo, mi hermano, quién con nuestra peculiar manera de relacionarnos, siempre sentí su apoyo, especialmente en los momentos difíciles.

Además un reconocimiento especial a todos mis amigos y amigas que me acompañaron desde el primer momento que entré a la Universidad. Los amigos vitalicios que formé de la querida Sección 6 del 2009, además de todos mis compañeros y compañeras de Computación, que me aceptaron tal como soy, y que me apoyaron y ayudaron en cualquier necesidad que tuviera.

Finalmente agradecer a profesores que me inculcaron la pasión por la Ingeniería y la Computación, en especial a mi Profesor Guía, Alejandro Hevia, que pese a las dificultades iniciales, siguió apoyándome y brindando su ayuda para que pudiera desarrollar una memoria que me gustara, y que tuviera un resultado concreto.

Tabla de Contenido

Índice de figuras

Capítulo 1

Introducción

1.1. Votación Electrónica

La votación electrónica consiste en un sistema de votación que utiliza mecanismos computacionales, tanto para emitir votos, como para contarlos. Existen diversas propuestas de como llevar esto a cabo, las cuales se diferencian en las características de la votación a privilegiar (usabilidad, bajo costo de implementación, tecnología de punta, reusabilidad, etc.). Entre estas propuestas se encuentran sistemas que utilizan papeletas de votación leídas por escáneres ópticos, emisión de votos vía un computador ubicado en el recinto de votación o emisión de manera remota a través de Internet, con conteos totalmente electrónicos o con conteos donde se manipula una versión física del voto, entre otros.

La votación electrónica se ha propuesto desde los años '60. Se han diseñado muchas variantes de técnicas y mecanismos, tanto para la emisión de los votos, como para el conteo de éstos. La gran mayoría de estas variantes son de carácter "académico", es decir, no se han plasmado en sistemas de uso masivo. Sin embargo, existen varios países que tienen funcionando actualmente algún tipo de votación electrónica para realizar sus actos democráticos habituales. Independiente de cómo se lleve a cabo la votación, ésta debe preservar condiciones básicas que permitan la validez de los resultados. Entre estas condiciones están: integridad de la votación, anonimato de los votos, resistencia a coerción y transparencia del proceso, entre algunas más.

Al momento de analizar si un sistema de votación es bueno o no, se ven una serie de propiedades que debe cumplir, principalmente relacionadas con el contexto en donde se desarrollará la votación (qué es lo que se elige, lugar de la votación, número de votantes, implicancias socio-políticas, etc.). Entre estas variables a analizar están la transparencia, tecnología utilizada, garantías de seguridad, posibilidad de auditoría del proceso, especificación del software y hardware utilizado, entre algunas más.

Las ventajas de poseer un sistema electrónico de votación dependen mucho del contexto donde se utilizará. En varios casos podría añadir complejidades innecesarias que hacen que el sistema sea menos usable. En la gran mayoría, sin embargo, un sistema electrónico añade

mejoras sustanciales, por ejemplo, mayor accesibilidad al voto por parte de votantes con discapacidades físicas. Esto es implementar soluciones para que personas con discapacidad, y que no pueden emitir su voto de buena manera en un sistema clásico de votación, lo puedan realizar, sin la necesidad de dejar de lado alguno de sus derechos, como lo es principalmente la privacidad del voto. Además de esto, se acelera el conteo de votos en muchos casos, además que si se realiza de buena manera, se emplean técnicas matemáticas que pueden asegurar que el sistema sea altamente seguro.

Llevar a cabo un buen sistema de votación electrónica, implica un esfuerzo multidisciplinario, pensando en las consecuencias que implican si el sistema falla. Es por ello que se deben conjugar áreas como Computación, Diseño, Ciencias Políticas, Sociología, entre otras. En este trabajo sólo se abordara la vertiente computacional del proceso, pero siempre se tiene que tener en cuenta lo complejo del problema.

1.2. *Voter-Verified Paper Audit Trail* (VVPAT)

Una característica importante que debiera tener todo sistema de votación electrónica, es la verificación física de que la parte electrónica se condice con la opción seleccionada [20]. Una solución a esto es el uso de *VVPAT*, que consiste en dejar una copia en papel en el recinto de votación a la hora de emitir el voto electrónico. Es una manera de dejar constancia al votante de que en caso de falla del sistema electrónico, existen copias en papel ("a la antigua") del voto emitido, y éste será contabilizado apropiadamente.

Este es un "mecanismo de defensa" del sistema en general, para poder ser rescatado y ser aún válido si es que se llegaran a presentar sospechas de algún mal funcionamiento (principalmente que existen votos que no fueron contabilizados o fueron modificados).

Hay que destacar que utilizar *VVPAT* no hace dejar de lado la preocupación de que el sistema electrónico funcione, y es más que todo una salida de emergencia que tiene el sistema frente a algún imprevisto que no estaba pensado en un primer momento.

1.3. Experiencias Internacionales

Como se dijo anteriormente, han habido muchos países que han implementado distintos tipos de votación electrónica a lo largo de los años, con resultados distintos, pero con un denominador común: que la gran mayoría ha sido cuestionado de alguna manera, ya sea por supuestos ataques que se realizaron, o por su poca transparencia en el cómo funciona el sistema, o bien, brechas de seguridad que pudieron ser sido traspasadas. Entre estos ejemplos de sistemas cuestionados, están sistemas de votación utilizados en EE.UU. [17], Brasil [3], Holanda [18], Bélgica [8], India [25], Estonia [19], entre otros.

Los sistemas utilizados en los países anteriormente nombrados han sido ampliamente analizados e identificados sus pros y contras. Lamentablemente, según los estudios realizados,

la gran mayoría no ha funcionado de manera correcta, por lo que existen constantes cuestionamientos a su utilización. Varios estudios realizados han encontrado diversos problemas recurrentes en sus implementaciones, que hacen temer un fallido funcionamiento de los sistemas, tarde o temprano. Código fuente no público [14], pobre utilización de criptografía para preservar seguridad [6] y diseños poco transparentes, entre otros más, son algunos de las debilidades de varios de dichos sistemas.

Un problema importante que se ha evidenciado en los análisis anteriores, es que la ciudadanía desconfía mucho de la tecnología a la hora de utilizarla en votación electrónica. Eso, principalmente, por las malas experiencias que han tenido algunos sistemas de votación utilizados alrededor del mundo. Un ejemplo es el nombrado fallido sistema utilizado en Holanda [18]. El cual muestra que es muy difícil recuperar la confianza una vez que el sistema implementado fracasa. Es más, tanto en Holanda como en Alemania[1], ha sido prohibido el uso de máquinas electrónicas en los procesos eleccionarios de dichos países.

1.4. Justificación y Motivación

En Bélgica se realizó hace un par de años una modificación importante al sistema de votación utilizado [11], debido a que el antiguo había sido fuertemente cuestionado y había presentado muchas dudas acerca de su funcionamiento [8]. El nuevo sistema de votación tiene entre sus características el uso de un *VVPAT*, en donde los votos se guardan en una urna (en papel) en el recinto de votación. Esto implica que los votos puedan ser auditados de manera física, lo cual fue presentado como un requisito fundamental a la hora de modificar el sistema de votación. Esta medida se justifica por el hecho de que si existiera algún problema con la parte electrónica, aún se podría realizar un conteo de los votos. A pesar de ello, el sistema electrónico cuenta con protocolos de seguridad (métodos criptográficos, por nombrar a alguno), con lo cual no debiera existir la necesidad de recurrir al conteo de los votos depositados en la urna.

Existe otra propuesta similar que utiliza *VVPAT* junto con la votación electrónica [5]. En ella se propone un sistema dual, que significa que en la misma papeleta de votación se despliega la opción escogida, tanto en texto plano, como en forma encriptada para ser leída por un lector (parte electrónica). Con esto tenemos las dos caras de la moneda, donde la parte electrónica se registra en una memoria, y la parte con el texto plano se deposita dentro de una urna, de manera similar al sistema utilizado en Bélgica.

En ambos casos, se evidencia un problema que motiva este trabajo. El votante no tiene como saber que la parte del texto plano y la parte encriptada tienen relación. Están juntas, pero pudieron haber sido creadas de manera independiente, por lo que resulta necesario poder brindarle al votante la posibilidad de poder verificar que la encriptación corresponde a su selección, para así poder ofrecer una mayor seguridad y confianza del votante hacia el sistema de votación.

[1] `http://www.trustthevote.org/electronic-voting-banned-in-germany`

1.5. Objetivos

1.5.1. Objetivo General

Diseñar e implementar un Sistema de Votación Electrónica que utilice *VVPAT* como manera de auditar el proceso de votación, además de incorporar un paso intermedio de verificación, donde el votante se preocupa de corroborar que la parte electrónica (encriptada) de su voto corresponde a la selección realizada que aparece en el *VVPAT*. Con esto, el votante tendrá confianza de que el registro electrónico de su voto corresponde a la selección que ha escogido.

1.5.2. Objetivos Específicos

- Diseñar un esquema de votación que posea *VVPAT* y que el votante pueda verificar que la encriptación de su selección se hizo correctamente, especificando las distintas acciones a realizar por un votante a medida que va siguiendo las etapas señaladas en el diseño.
- Implementar dicho sistema de votación, teniendo en cuenta que cada una de sus partes sea lo más simple posible, para así poder adaptar el sistema a una gran variedad de contextos distintos en donde se pueda llegar a utilizar.
- Procurar la simpleza del sistema para ser fácilmente auditable de manera pública, es decir, que un observador externo pueda revisar cómo funciona el sistema y quede poco margen para cuestionamientos. Esto implica una implementación con código de libre acceso, junto con la utilización de un hardware reducido y específico en su función.

1.6. Organización del Documento

A grandes rasgos, en este trabajo se describe la propuesta de un protocolo específico para llevar a cabo una votación electrónica, mostrando tanto su diseño, como una implementación realizada. En el Capítulo 2 se describen los antecedentes necesarios para comprender el protocolo propuesto, incluyendo las bases matemáticas (criptografía) utilizadas para la preservación de la seguridad del sistema, y la explicación de los trabajos relacionados que inspiraron este proyecto. Posteriormente en el Capítulo 3 se describe en detalle el protocolo, centrándose en su diseño y la justificación de cada una de sus partes. Luego, en el Capítulo 4 se describe la implementación del protocolo que se llevó a cabo, describiendo las tecnologías y las consideraciones a la hora de llevar a cabo el diseño presentado. En el Capítulo 5 se dan a conocer los resultados de variadas experimentaciones que se realizaron con el proyecto, sobre todo en temas de usabilidad y comprensión por parte de distintos usuarios. En el Capítulo 6 se brindan las posibles mejoras que puede tener tanto el protocolo mismo como parte de su implementación, en pos de una constante mejora de éste, pensando en poder utilizarlo en experiencias reales de votación. Finalmente en el Capítulo 7 se presentan las conclusiones de este trabajo y las experiencias que dejó su implementación y las pruebas realizadas.

Capítulo 2

Antecedentes

2.1. Sistema Criptográfico *Paillier*

2.1.1. Encriptación Asimétrica

Encriptar un mensaje significa modificarlo de tal manera que, cualquier observador distinto del receptor original (llamado 'adversario' de aquí en adelante), no pueda comprenderlo, y que solamente con el uso de una "llave" preestablecida (clave) sea posible descifrar (desencriptar) el contenido original del mensaje.

Antes de explicar como llevar a cabo este objetivo, es bueno tener claro algunos conceptos que se utilizarán:

- Mensaje (M): contenido que se quiere transmitir, pero que no sea posible ver o modificar por una entidad externa a la cual se le quiere enviar.
- *Alice* (A): se le llama comúnmente *Alice* a la persona que emite el mensaje.
- *Bob* (B): de la misma manera, a la persona que recibe el mensaje se le llama *Bob*.
- Adversario (`Adv`): cualquier persona o entidad que no sea *Alice* ni *Bob*, es decir, que no es destinataria del mensaje enviado.
- Encriptar un mensaje: modificar un mensaje, de tal manera, que sea ininteligible para cualquier `Adv`. A este mensaje modificado se le llama texto cifrado.
- Desencriptar un mensaje: al recibir un mensaje encriptado, modificarlo de tal manera que lo resultante sea el mensaje original enviado por *Alice*.
- Esquema de Encriptación: tupla de algoritmos $(\mathcal{K}, \mathcal{E}, \mathcal{D})$, que definen la manera de crear las claves, encriptar un mensaje y desencriptar un mensaje, respectivamente.

Para poder realizar este propósito, es decir, ocultar un mensaje para que éste sea indescifrable para cualquier `Adv`, existen principalmente dos protocolos. Estos se diferencian en la manera de utilizar las claves.

1. Encriptación Simétrica: se usa la misma clave k, tanto para encriptar como para des-

encriptar el mensaje. Esto quiere decir, que si *Alice* quiere enviarle un mensaje a *Bob*, deben "pactar" de antemano una clave compartida k_1. Por otro lado, si *Alice* quiere enviarle un mensaje a *Charlie*, ellos dos deben de poseer otra clave compartida k_2 para poder comunicarse. Como éste protocolo no es el utilizado en este trabajo, no se ahondará más en su explicación.

2. Encriptación Asimétrica (clave pública): en este protocolo se manejan dos claves: pk y sk.
   - pk (public key): esta clave es la utilizada para poder encriptar un mensaje. Si *Alice* quiere enviar un mensaje a *Bob*, éste último debe generar tanto pk como sk. *Alice*, para encriptar un mensaje M, utiliza la clave pk, que es pública, es decir, que cualquier persona que quiera enviar un mensaje secreto a *Bob*, debe encriptarlo utilizando la clave pk disponible.
   - sk (secret key): *Bob*, luego de recibir un mensaje encriptado (bajo la clave pk), debe desencriptarlo utilizando la clave sk asociada a pk. Hay que recalcar que sk debe mantenerse de manera privada (secreta y conocida solo por *Bob*), ya que de lo contrario cualquier persona (en especial un Adv.) puede desencriptar los mensajes enviados hacia *Bob* y conocer su contenido.

   De esta manera, al utilizar encriptación asimétrica, no es necesario que cada par de personas que se quieran comunicar deban establecer de antemano una clave compartida. Solo basta que los receptores de los mensajes generen un par de claves (pk, sk). Luego de esto, expongan de manera pública cada pk y guarden en secreto la correspondiente sk. Posteriormente, cualquier persona que se quiera comunicar con otra, debe buscar la clave pública de dicha persona (la receptora del mensaje), encriptar el mensaje y enviarlo. Por construcción, solo la persona con la correcta clave privada sk podrá desencriptar el mensaje.

En resumen, un esquema de encriptación asimétrica (de clave pública) consiste en tres algoritmos: $(\mathcal{K}, \mathcal{E}, \mathcal{D})$

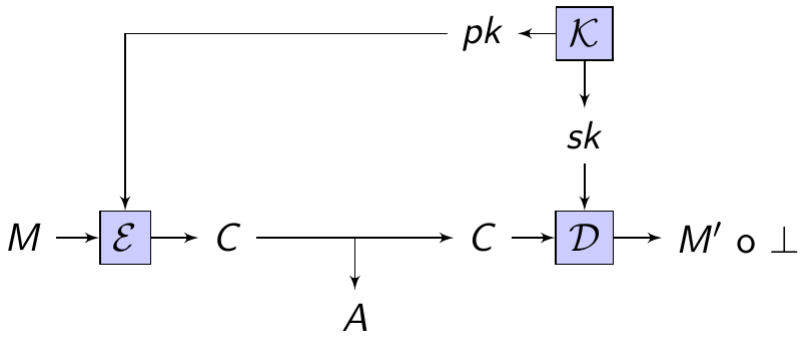

Figura 2.1: Esquema de Encriptación asimétrica

Para poder lograr el propósito de la encriptación asimétrica, existen varios esquemas conocidos y su seguridad testeada lo largo de los años. Entre estos podemos mencionar: *ElGamal*, *RSA* y *Paillier*. En este proyecto se utilizó el esquema de *Paillier*, debido a una

propiedad notable que posee, la cuál es la encriptación homomórfica, que será explicada a continuación.

2.1.2. Encriptación Homomórfica

Hay ocasiones que lo que se quiere finalmente no es desencriptar un mensaje en particular, sino que interesa realizar alguna función sobre el total de mensajes encriptados. Más específicamente, en el caso de una votación, a uno no le importa saber los votos individuales, en cambio lo que se busca, es cuántos votos consiguió cada candidato al final del día, es decir, la suma de estos votos. Existen esquemas de encriptación que permiten calcular dicha función sobre el total, sin la necesidad de desencriptar los mensajes uno a uno. A dicho esquema, se dice que posee una propiedad homomórfica.

En particular, una función homomórfica se define como una función $f(\cdot)$ que posee la siguiente propiedad:

$$f(a \cdot b) = f(a) * f(b)$$

done $\cdot$ y $*$ son operaciones sobre el $Dom(f)$ y $Rec(f)$, respectivamente.

En el caso del esquema *Paillier*, esta propiedad homomórfica (como será explicado más adelante) se aplica donde $*$ es la operación multiplicación y la operación $\cdot$ es la suma. Concretamente:

Sea $\mathcal{AE} = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ el esquema *Paillier*. Entonces resulta la siguiente propiedad:

$$\mathcal{D}(\mathcal{E}(m_1) \cdot \mathcal{E}(m_2) \cdot \ldots \cdot \mathcal{E}(m_n)) = m_1 + m_2 + \ldots + m_n$$

Esto quiere decir, que si utilizamos *Paillier* para encriptar votos, no es necesario que se desencripten uno por uno, sino que solo debemos multiplicar las encriptaciones, y al desencriptar dicho resultado, obtendremos la suma de los votos particulares.

2.1.3. Criptografía Umbral y Desencriptación Parcial

Otra propiedad importante que posee *Paillier*, es su capacidad de poder utilizar criptografía umbral en la desencriptación de los mensajes. La criptografía umbral consiste en la capacidad que posee un esquema de encriptación en poder repartir la clave privada sk en varios participantes, utilizando un protocolo denominado *secret sharing* [24], que consiste en poder dividir un secreto entre ℓ participantes, para posteriormente restituir el secreto utilizando solamente w partes ($w < \ell$). Un ejemplo sería que un cierto secreto s se divide en 5 partes: s_1, s_2, s_3, s_4, s_5, y posteriormente se necesitan sólo 3 partes cualesquiera, digamos s_1, s_4, s_5, para poder formar nuevamente el mismo secreto s.

Utilizando esta idea, junto con propiedades surgidas desde el esquema mismo de *Paillier*, es posible realizar una desencriptación umbral de un texto cifrado $c = \mathcal{E}(m)$. Para eso, cada participante calcula una desencriptación parcial del texto encriptado c, y luego todas las

desencriptaciones parciales son combinadas (en forma pública, por cualquiera) para obtener el texto plano m. En el caso de este trabajo, el mensaje m corresponde a la suma de los votos emitidos. Esto funciona como sigue:

1. Al principio de la votación, se genera el par de claves (pk, sk) para encriptar y desencriptar los votos, respectivamente. La clave pk (clave pública) se deja en los distintos dispositivos que la necesiten, y la clave sk se trata como un secreto en *secret sharing*, es decir, se divide entre ℓ participantes (autoridades), estableciendo un número w de ellos que pueden "reconstruir" la clave (en realidad esto nunca se realizará).
2. Se lleva a cabo la votación, donde cada voto va encriptado utilizando la clave pk (este protocolo será especificado en el Capítulo 3).
3. Al final del día se tienen todos los votos encriptados, los cuales son repartidos entre w autoridades dispuestas a utilizar su parte de la clave sk para revelar los resultados.
4. Cada una de las w autoridades, valiéndose de la propiedad homomórfica de *Paillier*, multiplica los votos encriptados, obteniendo la encriptación de la urna (llamémosle $\hat{c}$).
5. En vez de reconstruir sk, cada autoridad desencripta $\hat{c}$ utilizando solamente su parte sk_i. Luego de esto, entrega su desencriptación parcial a una entidad centralizada.
6. Esta entidad recibe cada una de las desencriptaciones parciales, y puede generar la desencriptación total de la multiplicación de los votos de manera pública, obteniendo el resultado total de la votación.

Al realizar este protocolo se asegura que la clave sk nunca es reconstruida, ya que cada autoridad nunca revela su parte que le corresponde. Además, no es posible desencriptar los votos uno a uno, sino que solo se revela el resultado total. La explicación del detalle de esta propiedad será explicada en la próxima subsección.

2.1.4. Esquema de Paillier

Para este trabajo se utilizó una variante del esquema de encriptación original de *Paillier*, el cual permite realizar la desencriptación umbral bajo el protocolo explicado anteriormente, manteniendo aún la seguridad y la propiedad homomórfica que caracteriza al esquema [16]. La gran mayoría de los esquemas de encriptación de clave pública basan su seguridad con respecto a un problema matemático difícil de resolver computacionalmente. En el caso de *Paillier*, este problema es calcular las clases del enésimo residuo (Residuidad Compuesta explicada en el *paper* original que propone el esquema [21]). Basándose en eso, a continuación se explicará cada uno de los algoritmos que representan la variante del esquema de *Paillier* que permite la utilización de desencriptación parcial:

- Generación de Claves ($\mathcal{K}$):
    1. Se generan dos números primos p, q que satisfacen $p = 2p' + 1$ y $q = 2q' + 1$, siendo p', q' números primos distintos de p, q.
    2. Se calcula $n = pq$ y $m = p'q'$.

3. Se decide un $s > 0$ tal que los textos planos son elementos del conjunto $\mathbb{Z}_{n^s}$[1]. Se escoge además un d tal que $d = 0$ mód m y $d = 1$ mód n^s.
4. Luego se construye la función polinomial $f(X) = \sum_{i=0}^{w-1} a_i X^i$ mód $n^s m$, escogiendo a_i $(0 < i < w)$ como valores aleatorios entre $\{0, \ldots, n^s \cdot m - 1\}$ y $a_0 = d$. Esta función se basa en el esquema *secret sharing* sobre como distribuir el secreto entre varios participantes.
5. La clave privada sk_i a repartir entre las distintas autoridades es $sk_i = f(i)$ para $1 \leq i \leq \ell$, donde ℓ es el número de autoridades. La clave pública resulta ser $pk = (n, s)$.

- Encriptación ($\mathcal{E}$):

  Para encriptar un mensaje M, se elige un número aleatorio $r \in Z_n^*$, y se calcula el texto cifrado $c = (n+1)^M r^{n^s}$.

- Desencriptación Compartida ($\mathcal{D}$):

  La autoridad i-ésima calculará $\hat{c}_i = c^{2 \cdot \ell \cdot sk_i}$, donde c es el texto cifrado.

- Combinar Desencriptaciones Parciales:

  Si tenemos las requeridas w (o más) partes para obtener el resultado final, las combinamos utilizando un subconjunto S de w partes y se calcula:

$$c' = \prod_{i \in S} \hat{c}_i^{2 \cdot \lambda_{0,i}^S} \text{ mód } n^{s+1} \quad ; \quad \text{donde} \quad \lambda_{0,i}^S = \ell \prod_{i' \in Si} \frac{i'}{i - i'} \in Z$$

  El valor c' tendrá la forma $c' = c^{4 \cdot \ell^2 \cdot a_0} = c^{4 \cdot \ell^2 \cdot d}$. Notando que $4\ell^2 d = 4\ell^2$ mód n^s, concluimos que $c' = (1+n)^{4 \cdot \ell^2 \cdot M r^{n^s}}$ mód n^{s+1}, donde M es el texto plano deseado. De aquí podemos obtener $4\ell^2 M$, aplicando una propiedad demostrada en [16]; luego esto se multiplica por $(4\ell^2)^{-1}$ mód n^s, obteniendo el texto plano M deseado.

En la referencia [16] se realiza un análisis acabado de la seguridad de este esquema, además de su correctitud con respecto a la obtención del texto plano requerido para mostrar un resultado correcto.

Haciendo la conexión entre el esquema anteriormente descrito, y este trabajo, cada voto lo vamos a representar por v_j. Estos votos son encriptados por el esquema de *Paillier*, transformándose en los textos cifrados c_j. Al final del día, cada autoridad i recibe los c_j, los multiplica, y ese resultado es el texto cifrado c. Lo desencripta parcialmente,generado $\hat{c}_i$. Finalmente, la entidad central combina los $\hat{c}_i$, obteniendo el texto V, que en este caso es la suma de los votos individuales, debido a la propiedad homomórfica de *Paillier*.

Con esto nos damos cuenta de la mejora sustancial que se logró: al no permitir la construcción de la clave sk, es imposible desencriptar los votos por separado, por lo que la opción de cada persona siempre quedará en secreto, pero aun así es posible rescatar el resultado final, que es lo que se espera al momento de montar una votación.

[1]El conjunto $\mathbb{Z}_j$ denota todos los números enteros mayores que 0, hasta j, y además $Z_j^* = \{a \in Z_j : mcd(a, j) = 1\}$

2.2. Esquema de Firma Electrónica *RSA*

2.2.1. Firma Electrónica

Un aspecto importante para preservar la seguridad no es sólo que un adversario no conozca el contenido de un cierto mensaje, sino que también el receptor pueda asegurar fehacientemente que el mensaje recibido (encriptado o no), viene de una cierta persona en particular, y no de alguien que se quiere hacer pasar por dicha persona. Esto se llama *autenticidad*, y al igual que en la encriptación, se puede resolver utilizando un protocolo de clave compartida o uno de claves pública y privada. El protocolo de clave compartida no es el utilizado en este proyecto, y por ende no se va a profundizar en su explicación.

Para asegurar autenticidad con un esquema de clave pública, se utilizan las *Firmas Digitales*, la cual tiene su símil en el esquema de encriptación asimétrico, pero se diferencia en que justamente no tiene como fin esconder un cierto mensaje, sino que asegurar que el emisor del mensaje es quien dice ser.

Un esquema de firma digital posee tres algoritmos: $\mathcal{DS} = (\mathcal{K}, \mathcal{S}, \mathcal{V})$, los cuales respectivamente generan las claves, firman el mensaje y finalmente verifican la autenticidad de la firma.

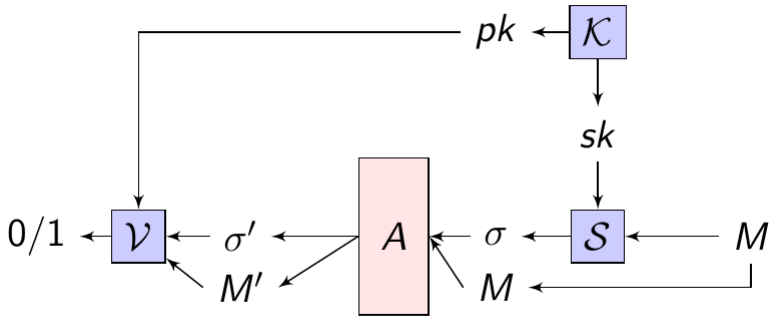

Figura 2.2: Esquema de Firma Digital

Los pasos a seguir para firmar y posteriormente verificar un mensaje son los siguientes (supondremos que *Alice* quiere enviarle un mensaje M a *Bob*):

1. *Alice* genera un par de claves utilizando $\mathcal{K}$, lo cual produce (sk, pk), que posteriormente mantiene en secreto sk y revela públicamente pk.
2. Al momento de enviar el mensaje M, *Alice* utiliza sk para generar la firma σ sobre M. Luego, envía el par (M, σ) hacia *Bob*.
3. Al recibir el mensaje, *Bob* (o potencialmente cualquier persona) puede verificar, utilizando pk, que la firma σ sobre M es verídica y pertenece a *Alice*.

En este proyecto se utilizó el esquema *RSA*, el cual está muy masificado, debido a su comprobada seguridad.

2.2.2. Esquema RSA

El esquema *RSA* fue propuesto en 1977 por Ron Rivest, Adi Shamir y Leonard Adleman [23]. Es un esquema de clave pública, que funciona tanto para encriptar mensajes como para producir firmas digitales. Como fue dicho anteriormente, estos esquemas residen su seguridad en la dificultad computacional de resolver cierto problema. En el caso de RSA el problema es la dificultad de factorizar un cierto número entero en sus factores primos.

A continuación se explicarán cada uno de los tres algoritmos que forman el esquema RSA de firma digital $\mathcal{DS} = (\mathcal{K}, \mathcal{S}, \mathcal{V})$:

- Generación de Claves ($\mathcal{K}$):
  1. Se eligen dos números primos aleatorios, de largos (en bits) similares (p, q) (en el largo de estos números radica la seguridad del esquema). Además se escoge un número e aleatorio (hay veces que la elección de este número no es aleatorio y típicamente se escoge $e = 3$ ó $e = 2^{16} + 1$).
  2. Se calcula $N = pq$, $M = (p-1)(q-1)$.
  3. Es necesario que $mcd(e, M) = 1$. Si no sucede esto, se vuelven a elegir un par (p, q) y se repite el proceso.
  4. Calculamos d, tal que $e \cdot d \equiv 1 \pmod{M}$. Llamamos a d el inverso modular de e.
  5. Se retorna $pk = (N, e)$ y $sk = (N, d)$
- Firma del Mensaje ($\mathcal{S}$):

  Si se quiere firmar el mensaje M, se calcula $\sigma(M) = \bar{M}^d \bmod N$, lo cual es la firma sobre una versión $\bar{M} = \text{Pad}(M)$ de M con *padding*[2].
- Verificación de la Firma ($\mathcal{V}$):

  Para verificar la firma, se necesita el par (σ, M), y se comprueba que $\sigma^e \equiv M \pmod{N}$. Si resulta de esta forma, la firma es válida, si no, no lo es y ha sido alterada en el camino.

La seguridad de *RSA* reside en el hecho que al recibir el número N es difícil poder computar su descomposición en números primos $p \cdot q$. Para tener más información sobre la correctitud del algoritmo, dirigirse a la referencia original de la propuesta [23].

¿Porqué utilizar firma en un sistema de votación? La firma se utiliza para poder comprobar la autenticidad de los votos, es decir, que fueron producidos por personas habilitadas para la votación. Además con esto, podemos obtener otras dos propiedades: verificar que no hayan votos duplicados (que una persona haya votado dos veces), además que las personas puedan verificar que su voto ha sido contabilizado. Lo que se realiza es que cada votante posee un par de claves (pk, sk) generados a través del esquema *RSA*. Luego, al momento de generarse el voto encriptado, se le solicita la clave privada sk al votante para que lo firme, es decir, el mensaje M es el voto encriptado con *Paillier*, y se genera una firma σ sobre dicho voto. Luego, antes de registrar el voto, se verifica dicha firma sobre el voto encriptado, utilizando la clave pública pk del mismo votante. Si se comprueba la veracidad, además de que no haya votado antes la misma persona, se procede a registrar el voto. Una mayor explicación de este

[2] `http://www.emc.com/emc-plus/rsa-labs/historical/raising-standard-rsa-signatures-rsa-pss.htm`

protocolo se realizará en el Capítulo 3, donde se explicará paso a paso lo ocurrido durante el proceso de votación.

2.3. Sistemas de Votación Electrónica relacionados

2.3.1. *Dual Voting System*

Existen varias propuestas de sistemas duales de votación, es decir, que combinan el voto en papel con un voto encriptado electrónico. En particular este trabajo se basó en una implementación realizada en el año 2012 por académicos israelíes e italianos [5].

El sistema tiene como punto de partida retener el *look-and-feel* de un sistema clásico de votación (basada en papel), integrando elementos electrónicos que hacen aumentar su seguridad, añadiendo verificación *end-to-end*, garantizados por el uso de herramientas criptográficas. El objetivo del sistema es poder correr dos procesos de votación en paralelo, uno basado en papel y el otro electrónico, dotando la consistencia necesaria entre ambos con las herramientas nombradas anteriormente.

En general, el proceso consiste en que cada votante, al momento de ejercer su voto, genera una papeleta con dos partes desprendibles una de la otra (llamada *dual-ballot*). En la primera se desplegaba la opción escogida en texto plano, la cual debía doblarse y depositar en una urna dispuesta en la mesa de votación. La segunda parte contenía un código QR[3] que representaba la opción escogida, encriptada y firmada electrónicamente por el votante. Esta segunda parte era leída por un lector que enviaba el voto a un servidor central. Posteriormente los votantes podían verificar en forma *online* que su voto estaba presente en el conteo final.

En el sistema existe una opción de poder auditar la máquina que genera las papeletas, pero ésta se encuentra escondida (hay que tocar una parte especial de la pantalla táctil). Los votantes si no conocían este procedimiento, quedan a oscuras con respecto a si la parte electrónica justamente refleja la opción que ellos seleccionaron.

Finalmente el proceso terminaba realizando un conteo público de los votos electrónicos, utilizando un *bulletin-board*. El conteo manual de los votos depositados en la urna, se realiza también de manera pública en el recinto de votación. La decisión de que resultado respetar (si es que el de la parte electrónica difiere con el de la votación manual) queda en manos de las autoridades de la elección, aunque la política siempre beneficiaba a la votación de los votos en papel.

Luego de varias pruebas y cuestionarios a los usuarios, los dos mayores problemas que se evidenciaron [5] fueron la usabilidad del sistema (en particular, como doblar el voto para esconder el texto de la opción escogida) y el proceso de verificación online. Más aún, lo que más llamó la atención fue la alta confianza que tuvieron los votantes hacia el sistema de votación, esto debido a que les gustó el hecho de ver su opción escogida en texto plano, como

[3] `http://www.qrcode.com/en/index.html`

solían hacerlo con un sistema estándar, lo que hacía saber que frente a cualquier sospecha de fraude o de mal funcionamiento de la parte electrónica, los votos físicos iban a estar ahí para poder realizar cualquier tipo de auditoría al proceso y terminar con un resultado íntegro de la votación.

Del análisis de esta implementación, los autores sacaron varias conclusiones. La primera es que el hecho de contar con un *VVPAT* claro y cercano, hace que la gente confíe mucho más en el sistema. Además la usabilidad del sistema es aún muy confusa como para masificar su uso. Por último, los votantes tenían la intención de verificar su voto, pero el proceso que se les brindó para ello resultó ser un tanto complicado. Un paso importante es trabajar en esto, para que se pueda tener una verificación clara y en el momento, de que la parte electrónica expresa la opción escogida por el votante.

2.3.2. Sistema de votación en Bélgica

Como fue dicho anteriormente el sistema de votación oficial en Bélgica fue modificado el año 2012 debido a variados cuestionamientos hechos al antiguo sistema [8]. En Bélgica se viene utilizando un sistema de votación electrónico desde el año 1991, aunque su cobertura ha llegado sólo hasta un 44 % de los votantes. Si bien el sistema funcionaba de manera correcta, los cuestionamientos se concentraban en la falta de transparencia que otorgaba el sistema, debido a la falta de un comprobante auditable sobre la opción escogida por los votantes (*VVPAT*). Variadas mejoras se fueron incluyendo al sistema, en pos de ir aumentando la transparencia del proceso. Entre estas mejoras se encuentran: publicación del código fuente del software utilizado, la creación de un Colegio de Expertos preocupado de monitorear la buena funcionalidad del sistema, además de la inclusión de un *ticket* (*VVPAT*) que permite a los votantes confiar en que habrá un respaldo de su voto electrónico. Como conclusión de estos procesos de mejora, se solicitó realizar un estudio comparativo de distintas alternativas de sistemas de votación, para poder modificar el que se utilizaba en la época, junto con empezar un debate a nivel nacional sobre cuál es la mejor alternativa para efectuar los sufragios. Producto del estudio y de un reporte realizado por el Consejo Europeo, se decidió adoptar el sistema que está actualmente en uso en Bélgica y que sirvió como base para este trabajo.

Este nuevo sistema [11] tomó las consideraciones hechas por varios estudios sobre qué características debiera tener para asegurar la confianza de sus votantes, además de preservar la seguridad e integridad de un proceso eleccionario.

El nuevo sistema incluye un *VVPAT* muy parecido al sistema dual, donde en una misma papeleta se encuentra en texto plano la opción escogida, además de un código *QR* con la misma opción, lo que hace aumentar la confianza de los votantes hacia el sistema de manera notable, como fue discutido anteriormente. De la misma manera que el sistema dual, el votante debe doblar la papeleta, escondiendo la opción escogida en texto plano. Luego de esto, la papeleta se deposita en una urna electrónica, la cual registra la opción electrónica y guarda la papeleta con el *VVPAT* al mismo tiempo. El sistema ya ha sido utilizado, y aun no se encuentran mayores problemas de seguridad y privacidad, por lo que es un buen caso para analizar.

Algunos aspectos negativos del nuevo sistema, son su poca (o nula) utilización de técnicas criptográficas sobre la parte electrónica del voto, lo que hace que puedan aparecer sospechas de un mal funcionamiento del sistema debido a la poca seguridad que pudiera tener. Un aspecto positivo y rescatable del sistema, es que cuenta con distintos módulos que poseen funcionalidades específicas que hacen que su complejidad sea más baja y sea mucho más fácil entender y auditar el proceso, ya que se puede separar en cada una de sus etapas. Los módulos son: una pantalla táctil donde se seleccionan los candidatos escogidos, una impresora de códigos, un escáner y una *urna electrónica* (en esta urna, se deja depositado el voto en texto plano, además de leer electrónicamente el código QR).

Con este sistema vemos que es posible una opción dual en un contexto importante (votaciones a nivel nacional en Bélgica), pero que aun hay problemas que resolver con temas de usabilidad y verificación. Nuevamente los votantes quedan a oscuras respecto a si su opción fue plasmada de buena manera en el código electrónico, sin la capacidad de verificarlo. Hay muchos incrementos en temas de usabilidad (con relación al sistema anteriormente descrito), pero que es necesario conjugar con técnicas criptográficas necesarias para preservar seguridad e integridad del proceso.

Capítulo 3

Sistema de Votación Dual y Modular

La propuesta de este trabajo consiste en un Sistema de Votación, que inspirándose tanto en el sistema utilizado en Bélgica, como el mostrado en la sección 2.3.1, utiliza la idea de *VVPAT* para tener un respaldo por si sucede cualquier problema o sospecha al sistema electrónico, además de incluir un módulo donde el mismo votante puede realizar una verificación de que el sistema electrónico va a registrar la selección que realmente quiere realizar. Otro aspecto importante para este proyecto, es el hecho que el sistema es altamente modular, es decir, cada operación y paso a seguir está separado de los otros, haciendo la auditoría más sencilla de realizar, además de dificultar la realización de cualquier ataque, ya que se deberían comprometer a más de una de las partes del sistema.

La explicación del sistema se realizará dividido en cada una de las partes que tiene que pasar un votante al momento de llegar al recinto de votación. Antes de esto, hay que tratar un problema ortogonal a este trabajo, que es la creación y obtención de firmas digitales para cada votante. Si bien es necesario para que el sistema de votación funcione, no es responsabilidad de éste ofrecer el servicio de creación de firmas, ya que debería ser labor de una entidad centralizada del Estado (en el caso de Chile, el Registro Civil), y de ahí el sistema de votación sacar provecho de esto. De igual manera este trabajo se hace cargo del tema, pero es bueno realizar esta aclaración. A continuación se procederá a explicar cada una de las partes del sistema.

3.1. Generación de Claves

En una primera oportunidad (suponiendo ya que cada persona posee su firma electrónica), es necesario generar las claves de la autoridad para la encriptación y desencriptación de los votos. Utilizando la variante del esquema de *Paillier* descrito anteriormente, se genera una clave pública pk, con la cual serán encriptados los votos generados posteriormente. Junto con esto, son generadas las partes sk_i $(1 \leq i \leq \ell)$, referentes a la repartición de la clave privada sk entre las distintas ℓ autoridades responsables (utilizando la criptografía umbral mencionada en el capítulo anterior). Luego de esto se deja de manera pública pk y cada autoridad se deja en secreto cada parte sk_i.

3.2. Primera Parte: Selección e Impresión del voto

El votante al llegar al recinto de votación, necesita identificarse en la mesa[1] (esta identificación *ex-ante* puede ser innecesaria, pero para conservar la cultura de los procesos eleccionarios, se incluye). Luego de la identificación, el vocal de mesa le señala entrar a una cabina disponible. Cada cabina posee dos máquinas, una de selección e impresión, y otra de verificación; esta última será explicada en la próxima subsección.

En la máquina de selección (que en este proyecto fue implementada con una *tablet*, pero esto no es imperativo), se despliegan todas las carreras y lista de candidatos por cada carrera que se votan en esa ocasión. En este momento, podría utilizarse el hecho que al llegar el votante a la mesa necesita identificarse, ya que puede que hayan personas que estén habilitadas para votar en ciertas carreras, y en otras no. Esto puede implementarse brindando al votante una *smart-card* que contenga la información de las carreras en las que el votante esté habilitado para participar.

Luego de que el votante realiza sus selecciones en las distintas carreras, que de aquí en adelante llamaremos v_j (j es el índice del votante), la máquina pregunta por una confirmación de las selecciones realizadas. El votante puede proseguir o volver atrás y cambiar lo que había seleccionado. Posteriormente, la máquina procede a encriptar el voto v_j utilizando el esquema de encriptación de *Paillier*, haciendo uso de la clave pública pk generada antes del inicio de la votación, junto con un número aleatorio r. Por lo tanto se genera:

$$c_j = \mathcal{E}(v_j, r)$$

Posteriormente a la encriptación, se le es solicitada al votante su firma digital sk_j, para poder verificar que la persona votando es quién dice ser (además posteriormente se verificará que no haya votado antes), de manera que se pueda verificar que su voto será tomado en cuenta al momento del conteo. Por lo tanto se forma la siguiente firma, utilizando el esquema *RSA* explicado anteriormente, además de un número $\bar{r}$ aleatorio:

$$\sigma_j = \mathcal{S}(c_j, sk_j, \bar{r})$$

Luego de realizar la firma, se prepara la papeleta de votación[2]. La papeleta de votación contiene 3 partes (ver Figura 4.3):

1. La primera parte contiene la opción escogida v_j en texto plano, convirtiéndose esta, en el futuro *VVPAT*.
2. En la segunda parte se despliega la encriptación del voto, junto con la firma sobre esta encriptación, es decir, el par (c_j, σ_j)
3. Por último, en la tercera parte se despliega la aleatoriedad utilizada para encriptar el voto, es decir, el valor r. Esto es necesario para posteriormente verificar que c_j realmente es la encriptación del valor v_j.

[1] La mesa de votación es el lugar donde se administra cada registro de los votos, y esta conformada por los vocales de mesa.

[2] Como el sistema utiliza *VVPAT*, es necesario depositar el voto en texto plano en una urna, por lo que se genera esta papeleta.

Después de generar esta papeleta, el votante selecciona que ésta se imprima en papel. La papeleta impresa tiene las 3 partes anteriormente descritas. Además éstas deben ser desprendibles una de la otra, teniendo una clara separación entre ellas. Otra consideración importante es que las partes 2 y 3 sean rápidamente leíbles por las distintas máquinas que siguen en el proceso de votación, por lo que en la implementación actual se utilizaron códigos QR para representarlos. Esto no es decisivo al momento de implementar el protocolo, pero sí debe realizarse de una manera tal que su lectura sea rápida y que se asegure su correctitud.

Luego de recibir la papeleta impresa, el votante pasa a la segunda máquina presente en la cabina de votación, que es para verificar la encriptación realizada. Los pasos seguidos en esta primera parte son resumidos en la Figura 3.1.

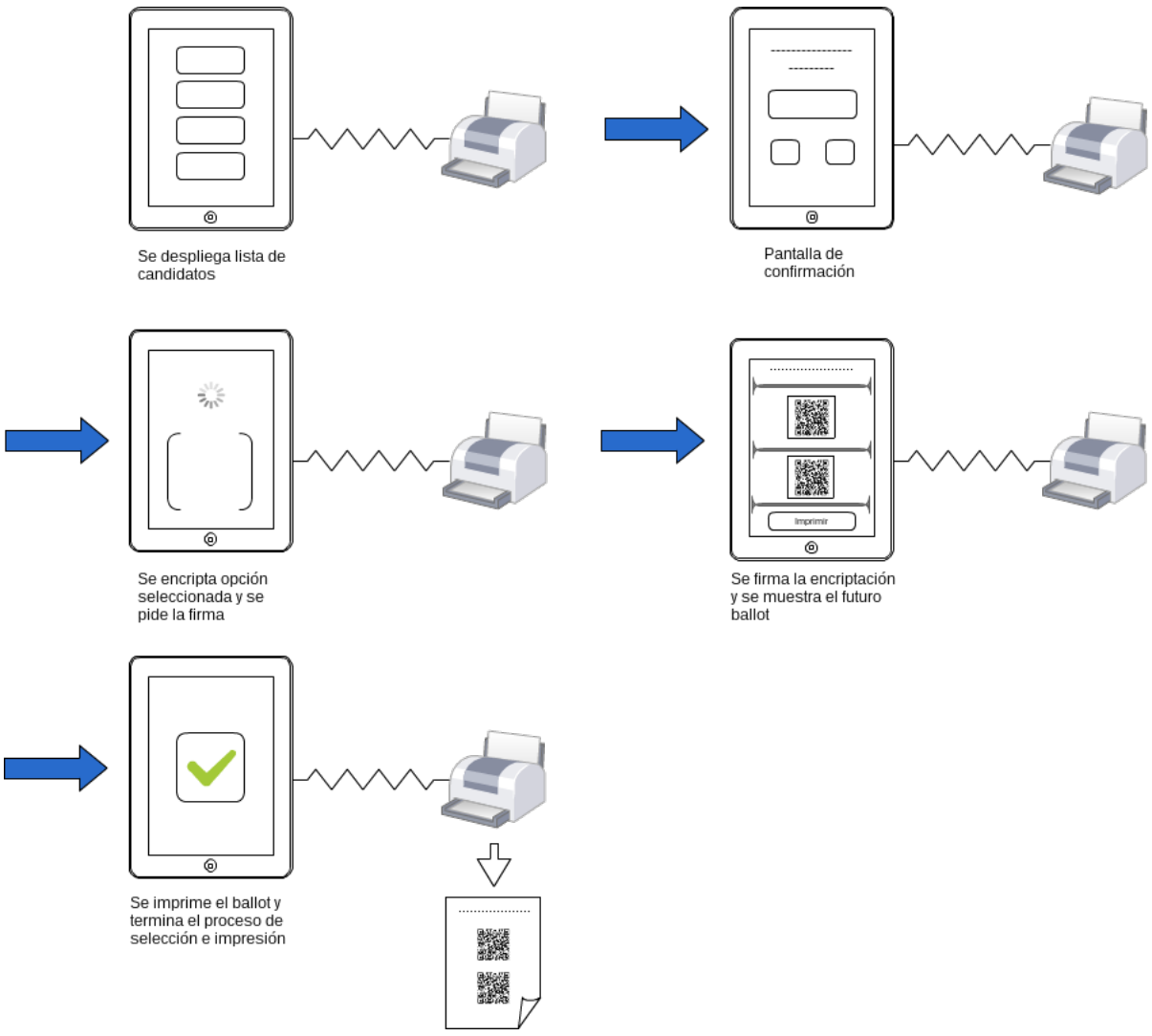

Figura 3.1: Primera parte: Selección e Impresión del voto

3.3. Segunda Parte: Verificación de la Encriptación

Luego de imprimir la papeleta, el votante pasa a operar con la segunda máquina dentro de la cabina de votación, el verificador de la encriptación. El propósito de esta máquina es convencer al votante que la segunda parte de su papeleta, la encriptación de su voto, contiene codificada su selección, y no a otro candidato. Esto sirve para prevenir que si la primera máquina fuese alterada, y estuviera encriptando otras opciones, y no las que el votante eligió, sea revelada su alteración.

Esta máquina posee de antemano todos las carreras y candidatos que se eligen en el día de la votación, además de la clave pública pk de la autoridad. Al iniciar su operación toma como entrada las partes 2 y 3 de la papeleta, es decir, el par (c_j, σ_j) y la aleatoriedad utilizada r. Procede con el siguiente algoritmo:

Algoritmo 1: Pseudo-código Algoritmo Verificador de Encriptación

```
Input: Encriptación c_j, aleatoriedad r
Output: Candidato v
Sea V conjunto de todas las opciones (candidatos) posibles;
v = none;
for v̄ ∈ V do
    Encriptar v̄ utilizando aleatoriedad r, c̄ = E(v̄, r);
    if c == c_j then
        v = v̄;
        break;
    end
end
```

Con esto, el algoritmo despliega la selección que está encriptada en la papeleta. Si el votante está de acuerdo con ello, es decir, si es la misma que aparece en la primera parte, debe doblar ésta para que no se revele, y salir de la cabina de votación. Si es que la opción desplegada por la máquina verificadora no es la misma que la que muestra la papeleta, quiere decir que la primera máquina operó de mala manera, por lo que el votante puede volver a ésta y realizar nuevamente la selección. De proseguir la inconsistencia entre lo desplegado por la verificadora y lo que aparece en la papeleta, debe ser un problema más serio que se encuentra en la primera máquina (sospecha de ataque al sistema) y debe comunicarse a las autoridades, quiénes deben decidir si cambiar la máquina u otra acción a tomar. Estos pasos seguidos en la segunda parte son resumidos en la Figura 3.2.

3.4. Tercera Parte: Autentificación y Emisión del voto

En este punto, se supone que el votante quedó satisfecho con lo que mostró la máquina verificadora (luego se hablará sobre los protocolos a seguir en caso contrario). El votante debe salir con la primera parte de su papeleta doblada, siguiendo las indicaciones impresas en la misma, para así poder esconder el texto que muestra su selección, pero dejando a la

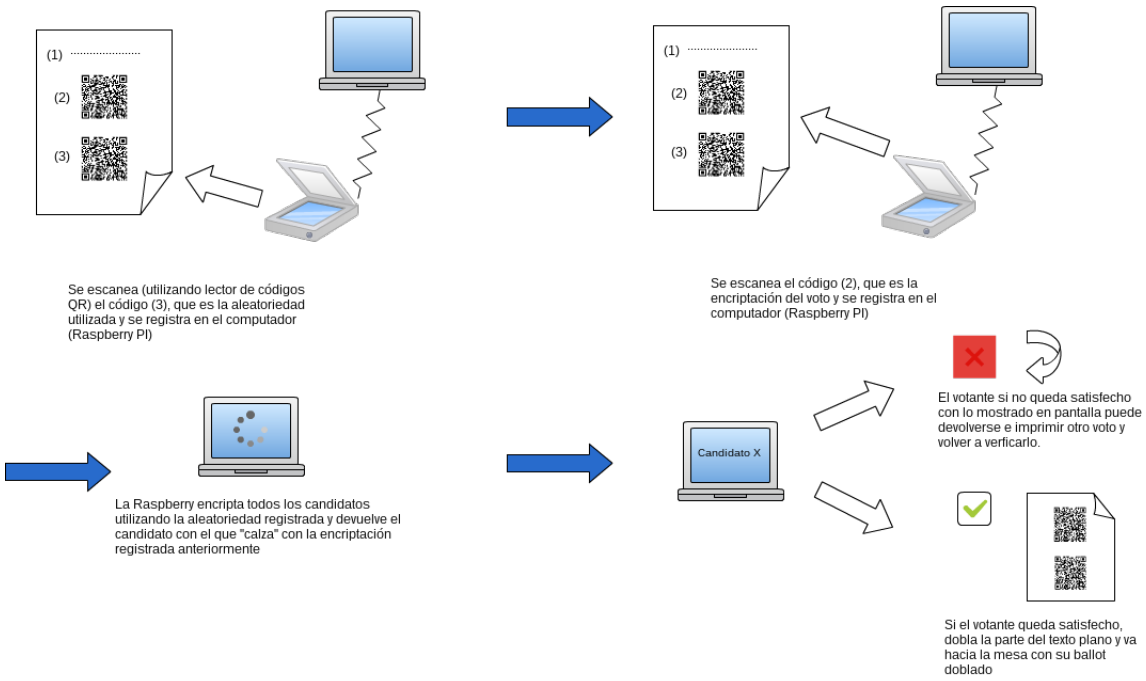

Figura 3.2: Segunda parte: Verificador de la Encriptación

vista los dos códigos QR que le siguen. Para una mayor seguridad se puede emplear un sello (estampilla) para impedir que se abra fácilmente la primera parte de la papeleta.

El votante se acerca a la mesa de votación con su papeleta, la cual es recibida por uno de los vocales, quién lo primero que realiza es destruir[3] la tercera parte de la papeleta (aleatoriedad r utilizada para encriptar), debido a que con ese valor el votante podría demostrarle a otra persona cual fue su selección[4]. Posteriormente se procede a registrar el voto electrónicamente, y esto se hace guardando la encriptación c_j del voto, previa verificación de la firma σ_j comparándola con la clave pública del votante (protocolo especificado en la sección 2.2.1). Si es que la firma es válida, además que se corrobore que el votante no haya votado anteriormente[5], se registra la encriptación c_j junto con la firma σ_j. Preferentemente hay que tratar de guardar la información en más de un sólo destino (por ejemplo, un par de pendrives o subir los votos a un servidor en la nube), para así asegurar la redundancia del voto por si sucede algún problema. Además, dependiendo de la implementación, puede ser subido a algún servidor público, que son conocidos como *bulletin boards*, donde se van registrando todos los votos que están siendo emitidos. Finalmente, se desprende la primera parte del voto (que está doblada, escondiendo la selección) y se deposita en una urna, tal como una votación común, asegurando que la elección posee un $VVPAT$ funcional y concordante con los votos registrados electrónicamente.

El votante se queda con la segunda parte de la papeleta (que fue registrada anteriormente),

[3]Debe procurarse de que sea muy difícil poder reconstruir este valor.

[4]Esta es una característica no deseada en un proceso de votación, conocida como privacidad fuerte del voto, que hace referencia a que el votante, aunque así lo desee, no puede demostrarle a otra persona cual fue su selección.

[5]En caso contrario, se le informa la situación al votante y no se registra su voto.

con la cual, si es que el sistema fue implementado con un *bulletin board* público, podría verificar que su voto va a ser contabilizado dentro del recuento al final del día. Todos estos pasos realizados en esta tercera parte son explicados en la Figura 3.3.

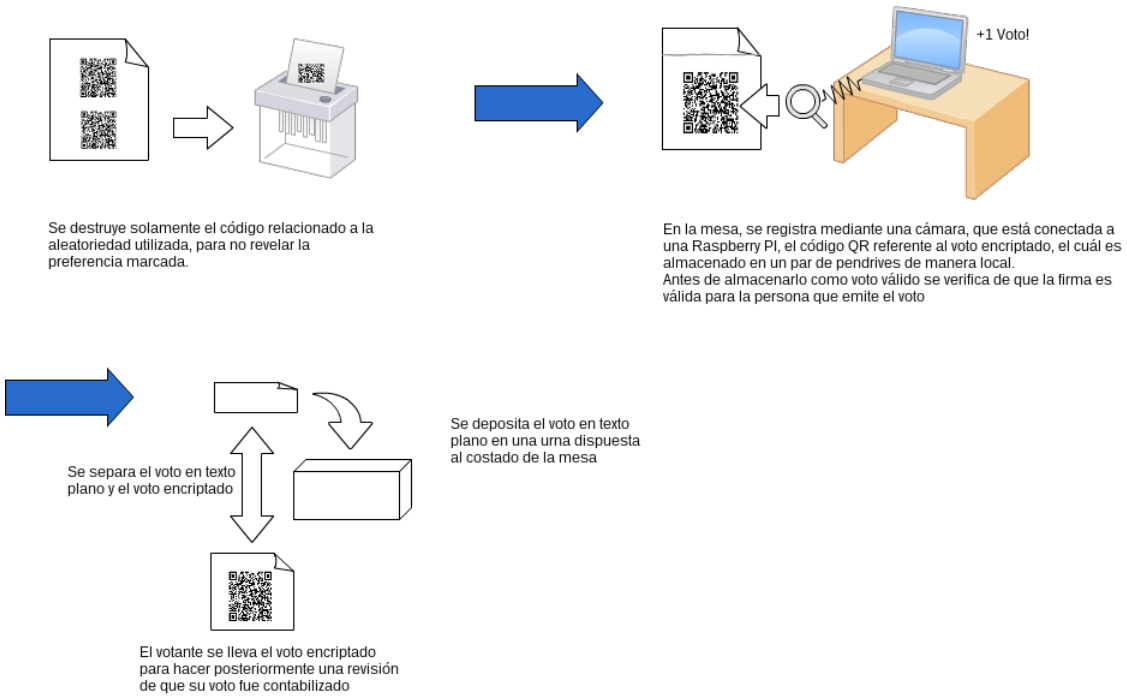

Figura 3.3: Tercera parte: Autentificación y Emisión del Voto

3.5. Conteo de los Votos

Al final del día se disponen todos los votos cifrados $c_j \in C$ $(1 \leq j \leq N)$, siendo N el número total de votos. Además se disponen de (al menos) w autoridades dispuestas a utilizar su porción de la clave privada sk_i para calcular el resultado total.

El conjunto C de los votos cifrados es público (por ejemplo, almacenados en un *bulletin board*), por lo que cada una de las w autoridades descarga u obtiene estos votos cifrados, los multiplica, y posteriormente calcula la desencriptación parcial utilizado su porción sk_i. Esto se realiza utilizando las propiedades de *Paillier* explicadas en el Capítulo 2. Luego, cada una de las autoridades envía su desencriptación parcial a una entidad centralizada.

Finalmente se combinan cada una de las w desencriptaciones parciales, mostrando el resultado final, que por la propiedad homomórfica de *Paillier*, corresponde a la suma individual de cada uno de los votos.

3.6. Aspectos Generales del Sistema

3.6.1. Modularidad

Una condición importante del diseño del sistema, es que cada una de las partes (selección, verificación y emisión de los votos) se realiza por separado, específicamente, en máquinas separadas, las cuáles no están conectadas una con la otra. Las máquinas se "comunican" mediante interfaces específicas: por ejemplo, la máquina de verificación recibe los valores de la encriptación y la aleatoriedad generadas en la primera máquina. Las ventajas de que el sistema sea modular, son varias, entre las cuales podemos mencionar:

1. Auditoría: al tener módulos específicos que desempeñan una cierta tarea, se hace más fácil auditar (verificar) que dicho módulo está haciendo el trabajo que se la ha encomendado de buena manera, debido a que este mismo trabajo es más acotado, por lo que las máquinas mismas son más simples en su construcción.
2. Múltiples proveedores: al poseer cierta independencia las máquinas unas de otra (excepto por la interfaz especificada por la que se "comunican"), invita a que puedan existir múltiples proveedores de éstas en el día de votación, en particular podrían disponerse de varias máquinas verificadoras. Todas las máquinas, eso sí, deben ser auditadas.
3. Disminución de la Superficie de Ataque: como cada uno de los módulos son altamente específicos en su función, esto implica que el código fuente que estén corriendo también se ve disminuido, por lo que la superficie de ataque se reduce considerablemente, implicando una mayor dificultad para el atacante de poder perpetrar una falla en el sistema. Por otro lado, si es percibido que se ha realizado un ataque a alguna máquina, se puede tener la posibilidad de modificar dicha máquina, lo cual no implicaría un cambio mayor en el proceso de votación, ya que la independencia de éstas hace que el cambio de una no repercuta en la necesidad de cambiar el resto. Esto también puede verse como una reducción de los gastos para manejar situaciones de emergencia.

3.6.2. Seguridad

Si bien desde un principio se especificó que no era objetivo de este trabajo realizar un estudio exhaustivo de la seguridad del diseño propuesto, sí es oportuno señalar algunos aspectos importantes en que se basa el sistema, y porque se menciona que la modularidad nos brinda la seguridad buscada, en especial, no poder vulnerar la privacidad del voto.

Existen dos máquinas que conocen la selección de una cierta papeleta, la primera y la segunda. La primera, ya que es ahí donde se realiza la selección, y la segunda porque logra desencriptar el voto. Lo importante es que ninguna de estas dos máquinas sabe quien es la persona que está realizando dicha selección. Si bien la primera máquina recibe la clave privada para realizar la firma, no sabe a que persona pertenece dicha clave, por lo que no puede realizar la asociación. La segunda máquina nunca solicita información al votante por lo que no puede tampoco realizar la asociación. La tercera máquina sí conoce la identidad del votante, ya que necesita verificar la firma del voto, pero no le es posible conocer la

selección, ya que no posee la clave correspondiente para desencriptar la opción. Por último, como se realiza una desencriptación umbral de los votos, nunca se reconstruye la clave privada necesaria para desencriptar los votos individuales, por lo que quedan en secreto para cualquier persona. Mencionar además que la persona que votó tampoco puede demostrar su selección, ya que tampoco posee la clave para desencriptar, o la aleatoriedad utilizada, ya que ésta se destruye al momento de registrar su voto.

Esta propiedad no se tendría si se concentran todas las operaciones en una sola máquina, por lo que es necesario separar las operaciones, y desprender de conexiones entre cada una de las máquinas, minimizando la superficie de ataque de un cierto adversario.

3.6.3. Dualidad

La dualidad del sistema hace referencia a la disponibilidad de un *VVPAT*, es decir, a la capacidad que tiene de poder entregar un resultado, a pesar de que exista una falla (o sospecha de ésta) en el sistema electrónico. Con esto, se dota de un "plan B" al sistema de votación. Además, los votantes poseen un respaldo claro de que su preferencia será respetada. Esto último debido a que el resultado electrónico podría ser contrastado con el resultado brindado por el *VVPAT*, al menos, durante las primeras veces en que se utiliza el sistema electrónico, para convencer a la población de que realmente funciona y empezar a brindar la confianza necesaria.

3.6.4. Preservar costumbres en la votación

Otro aspecto importante dentro del sistema, es su preservación de varios pasos a seguir que se han vuelto costumbre dentro de los procesos eleccionarios (hablando en el caso de Chile). Algunos de estos aspectos son:

- Autenticarse frente a la mesa de vocales al llegar al recinto.
- Pasar a marcar sus selecciones en una cámara secreta.
- Depositar un papel con su elección en una urna.
- Tener los resultados generales a poco tiempo de cerradas las urnas.

Además, recalcar que este sistema de votación es totalmente presencial, y no incluye aspectos de votación remota. Sí es importante mencionar que el votante puede revisar en línea si su voto fue contabilizado o no.

3.6.5. Manejo de errores

No existe un sistema computacional 100 % libre de fallas, por lo que es necesario poseer de antemano protocolos claros a realizar cuando se presenten algunos errores dentro del sistema. Éstos pueden ser internos del sistema (máquina de selección encripta de manera errónea los

votos, problemas de conexión eléctrica en algún momento, fallas en la red si es que se utiliza alguna comunicación con un servidor central, entre otras), como también producidos por los mismos votantes (mal orden en la lectura de los códigos al momento de verificar, firma mal utilizada, aparición de dos votos realizados por la misma persona, entre otros más). Obviamente los protocolos a realizar en cada una de las situaciones varían del contexto del proceso eleccionario que se esté desarrollando, pero lo importante es que estos estén explícitos y sean comunicados de antemano a los distintos participantes, para que no existan sospechas de ataques personales o desconocimiento de las reglas.

Si estamos en un contexto de elección de estudiantes, donde las implicancias político-sociales no son tan relevantes como en una votación nacional, además que el espectro de votantes están en una constante comunicación, no habría problemas en que los problemas internos se solucionen deteniendo las votaciones un cierto período de tiempo, hacer las verificaciones correspondientes de las máquinas, reemplazarlas si es que existe la posibilidad, y proseguir con el proceso. De no poseer capacidad de reemplazo, adoptar desde este momento en adelante un proceso de elección únicamente de manera manual, como un sistema clásico. Los errores producidos por los votantes, se pueden manejar pidiendo al votante que reitere el proceso de votación, guiándole claramente en los pasos a seguir y procurando que realice de manera correcta todos los pasos. Se hace la distinción con respecto a votaciones de nivel nacional, ya que detener el proceso eleccionario tiene muchas más repercusiones y los protocolos para realizar esto están altamente limitados.

Un error posible, independiente del contexto de la votación, es la falla en el momento de verificar la firma del votante. Esto se puede deber a: el votante no es quien dice ser y está utilizando la clave de otra persona, la máquina de selección realizó una mala lectura de la clave y produjo una firma errónea u otro problema de índole similar. En cualquier caso, como la verificación de la firma se realiza al final, es necesario que el votante realice nuevamente todo el proceso de votación, implicando un problema de usabilidad, debido a que se deberían reiniciar todos los pasos realizados. Una opción para subsanar este problema, es incluir la verificación de la firma en el segundo paso, es decir, al momento de verificar la encriptación del voto. Con esto, dejamos que se realicen en un solo lugar todas las posibles fallas del proceso, haciendo que bajo cualquier problema, solo se vaya un paso más atrás, y asegurando que el votante, al momento de salir de la cámara secreta, no va a tener que realizar todos los pasos nuevamente. Si bien se gana en usabilidad, se incluye un punto muy peligroso dentro del sistema, ya que la segunda máquina sabrá, en un cierto momento, el voto asociado a una cierta persona, ya que desencripta la opción seleccionada, además de tener que verificar la firma de esa persona. Esta situación no se daba al verificar la firma en el último paso. Si es que podemos asegurar que la segunda máquina no guarda estado, además de poseer solamente una memoria *read-only*, podríamos incluir la verificación de la firma en este paso. De lo contrario, no se puede, ya que se pondría en jaque la privacidad del voto, y se debe sacrificar un problema de usabilidad por poseer una característica fundamental dentro de un sistema de votación (privacidad).

3.6.6. Consideraciones sobre implementaciones

No existen restricciones a las distintas implementaciones que se realicen del sistema en cuestión, solamente el hecho de que las interfaces entre cada una de las máquinas estén explícitas y que sean el pilar fundamental por sobre se construyan las distintas aplicaciones. Algo muy deseable que se busca con este sistema, es que sea fácilmente replicable en variadas situaciones, por lo que los requisitos de hardware para la implementación deben ser mínimos, o en su defecto, bien específicos.

Junto con esto, para sacar provecho de la modularidad del sistema, el código de las distintas aplicaciones debe ser bien conciso y concreto, para poder facilitar el proceso de auditoría. Por otro lado, el hardware también debe ser bastante minimal, en concordancia con lo específico de las tareas que cumple cada aplicación.

Capítulo 4

Implementación realizada

En este capítulo se describirá la implementación que se realizó del sistema de votación, la cual tuvo como principal objetivo realizar un sistema minimal, que sea fácilmente auditable, con un código de fácil comprensión, además de utilizar tecnologías cercanas a la gente, en pos de una mayor usabilidad del sistema en general. Se describirá en el mismo orden en que se detalló el diseño del sistema.

4.1. Generación de Claves

Se generaron dos aplicaciones para generar claves, una para la generación de las firmas digitales bajo el esquema *RSA*, y otra para la generación de las claves para encriptar y desencriptar los votos bajo el esquema de *Paillier*.

La primera, denominada *RSAKeyGenerator*[1] es una aplicación desarrollada en *Java*[2], haciendo uso de el paquete `java.security`[3], el cual brinda todas las funcionalidades para poder generar el par de claves *RSA* que se necesita para obtener una firma digital. Junto con esto, como es una costumbre en todo el sistema, se genera la clave privada sk en forma de código QR, para facilitar su transporte, además de su lectura (electrónica). La clave pública pk es codificada en un archivo, el cual es repartido posteriormente a las otras máquinas que necesitan este valor para autentificar al votante. Esta aplicación esta hecha para ser manejada por un administrador del sistema de votación, o mejor dicho, del sistema de registro de los votantes, y no por cualquier usuario.

La segunda aplicación, denominada *ElectionKeysManager*[4] es la encargada de generar las claves para encriptar y desencriptar los votos, utilizando el esquema de *Paillier* previamente descrito. Para ello, se basa principalmente en una implementación de la variación de *Paillier*, perfilada hacia la desencriptación umbral, realizada por la *University of Texas at Dallas*, de-

[1] `https://github.com/CamiloG/RSA-Key-Generator`
[2] Todas las aplicaciones fueron programadas en *Java*
[3] `https://docs.oracle.com/javase/8/docs/api/java/security/package-summary.html`
[4] `https://github.com/CamiloG/Election-Keys-Manager`

nominada *UTD Paillier Threshold Encryption Toolbox*[5], que está inspirada en la descripción realizada en [16]. Al igual que la aplicación anterior, está orientada hacia los administradores de la votación, quiénes previamente al comienzo de ésta, deben ejecutar este programa para especificar las claves, además del número de autoridades ℓ a los cuáles repartir las claves, junto con el número w mínimo de partes necesarias para obtener la desencriptación (que se realiza de manera parcial como fue descrito en el capítulo anterior). Esta aplicación y la biblioteca externa están completamente desarrolladas en *Java*. El resultado del programa es una codificación de la clave pública, la cuál es repartida en todos los otros programas que la utilicen, además de ℓ archivos, conteniendo cada uno de ellos la parte de la clave privada que le toca a cada autoridad. Estos archivos deben ser repartidos a cada uno de las personas responsables (autoridades).

```
[camilo@asus 01 - ElectionKeysManager]$ java -jar ElectionKeysManager.jar
Bienvenida(o) a la generación de claves para la autoridad de la votación

NOTA: La generación de la clave privada se realiza a través de criptografía umbral
Ingrese el numero de partes (autoridades) en que se dividirá la clave privada: 6
Ingrese el numero minimo de partes (autoridades) que son necesarias para revelar la clave privada: 4
Generating p and p1
Trying prime
Finally a good pair
Generating q and q1
Trying prime
Finally a good pair
Generating d
Generating v
p :3085970189645679527919632314251509484 59
p1:1542985094822839763959816157125754742 29
q :3104898485549531114139801825276029810 03
q1:1552449242774765557069900912638014905 01
d :1142566103704026045971028633505339552402110707134122914448278891044573460757218158373381221980896243106625 1
198248308075659811911516530945804940139606065 91
v :1221225162540728026437866128812658578525409936086751816205515875266631129617245867240352349586573778398021 1
857340179076911896172725886642781485535389409 53

Repartir valores publicos guardados en publicValues/
Repartir partes de la clave privada entre las distintas autoridades, guardados en partsOfPrivateKey/

Proceso finalizado exitosamente.
```

Figura 4.1: Screenshot de Aplicación *ElectionKeysManager*

Una apreciación importante con estas dos aplicaciones, es que en el momento en que se ejecuten, debe existir un protocolo que elimine cualquier rastro de ambas claves privadas que se generan (tanto en el caso de la firma digital, como en el caso de *Paillier*), ya que si bien estas son entregadas a las personas responsables, la aplicación no realiza un borrado de los archivos que quedan en los equipos que corrieron estas dos aplicaciones. Por último, no hay mayores especificaciones o requisitos a los equipos para ejecutar estas aplicaciones, sólo que cuenten con *Java*[6] para poder ejecutar los archivos *.jar* que se generan.

4.2. Primera parte: Selección e Impresión del Voto

La primera aplicación que posee interacción con el votante es la máquina que selecciona e imprime la papeleta. Para una mayor cercanía con el votante, se ha decidido en esta

[5] `https://www.utdallas.edu/~mxk093120/paillier/`

[6] Para la experimentación se utilizó *Java 1.8* para todas las aplicaciones.

implementación realizar una aplicación que corra en una *tablet*[7]. Para ello se implementó una aplicación para *Android*, denominada *BallotSelection*[8], pensando además en la masificación que ha tenido este sistema operativo en los últimos años.

4.2.1. Requisitos Mínimos

Antes de utilizar la aplicación es necesario chequear la versión de *Android* instalada, además de algunos requisitos de aplicaciones y hardware disponible en la *tablet*:

- *Android* v.4.4 o superior: seleccionado porque que desde esta versión se integró un paquete para poder administrar una impresora desde la API.
- Aplicación *Barcode Scanner*[9]: necesaria para leer la firma digital del votante. Se utiliza esta aplicación en particular ya que posee una API bastante completa y fácil de usar[10].
- Cámara: necesaria también para leer la firma digital.

Importante recalcar que para que el sistema funcione bien, y no posea brechas de seguridad, es necesario que la *tablet* posea conexión a una impresora de manera directa (cable USB) y que no sea a través de una red. Lamentablemente para esta implementación eso no pudo ser posible, debido a que *Android* no posee dicha funcionalidad de manera nativa, y para hacerla funcionar hay que editar los controladores existentes actualmente, y se decidió no realizarlo debido al tiempo que tomaría esta modificación, y era necesario poder montar pruebas dentro del tiempo que se tiene para realizar este trabajo.

4.2.2. Paquetes y bibliotecas externas utilizadas

La aplicación implementada fue llamada *BallotSelection* y como fue explicado anteriormente, se desarrolló para la versión 4.4 de *Android*. Esta versión es la primera en brindar una API nativa para el control de impresoras y enviar a imprimir una cierta imagen. Esto es de vital importancia, ya que sin esto, no se podría implementar el *VVPAT* necesario para que el sistema sea robusto. Específicamente esto se describe en el paquete `android.print`[11].

Por otro lado, algo necesario también es el uso de una biblioteca externa para manejar la lectura de códigos *QR*, que se utiliza para leer la firma digital del votante necesaria para firmar el voto. Esa biblioteca es la provista por *ZXing*[10], la cual, como fue explicado anteriormente, provee una aplicación misma para la lectura de códigos *QR*, junto con la API necesaria para manejar lo que se lee.

[7] Se utilizó una Tablet Samsung Galaxy Tab 4 de 10.1¨ y S.O. Android 4.4.2
[8] `https://github.com/CamiloG/ballotSelection`
[9] `https://play.google.com/store/apps/details?id=com.google.zxing.client.android`
[10] `https://github.com/zxing/zxing`
[11] `https://developer.android.com/reference/android/print/package-summary.html`

4.2.3. Flujo de la aplicación

Al comenzar la votación, el votante debe seleccionar el ícono "Iniciar Votación". En la primera pantalla se despliega la lista de candidatos, tomada desde el archivo `candidates.xml`. Para esta implementación y las pruebas correspondientes, se realizó solamente una carrera, es decir, una pregunta, por lo que el archivo `candidates.xml` está hecho para solamente una pregunta. Al desplegarse la lista de candidatos, cada uno de ellos es un botón, con clara separación uno del otro, como se muestra en la Figura 4.2a.

(a) Lista de Candidatos (b) Pantalla de Confirmación de la Selección

Figura 4.2: Screenshots de *BallotSelection*

Luego de que el votante escoge su opción, se despliega la ventana de confirmación, la cual sólo muestra la opción que escogió el votante, además de botones que confirman o cancelan la selección. Si cancela la selección, se mueve a la pantalla anterior donde se muestra la lista de candidatos; si confirma, se muestra una ventana de aviso, explicando que a continuación se deberá acercar la firma digital (código *QR*) a la cámara para firmar el voto. La Figura 4.2b muestra la pantalla de confirmación.

Luego de hacer acuso de haber leído el aviso, la aplicación encripta la opción seleccionada utilizando la biblioteca externa de *Paillier* que fue introducida anteriormente (esto se realiza sin mostrar nada al votante). Luego de encriptar, se inicia la aplicación *Barcode Scanner*,

que consiste en abrir la cámara, y espera a que un código *QR* sea acercado a ésta. En este momento el votante debe acercar su clave privada (de su firma) a la cámara y esperar que sea leída. Al momento de ser leída, se emite un pequeño sonido, y la aplicación procede a firmar la encriptación realizada un paso antes. Luego de esto, se despliega en pantalla la papeleta que será posteriormente impresa, la cual está dividida en 3 partes: la primera con la opción seleccionada en texto plano, la segunda con un código *QR* representando la encriptación del voto y la firma sobre éste, y la última parte conteniendo la aleatoriedad usada, también en forma de código *QR*. Esto se ve en la Figura 4.3.

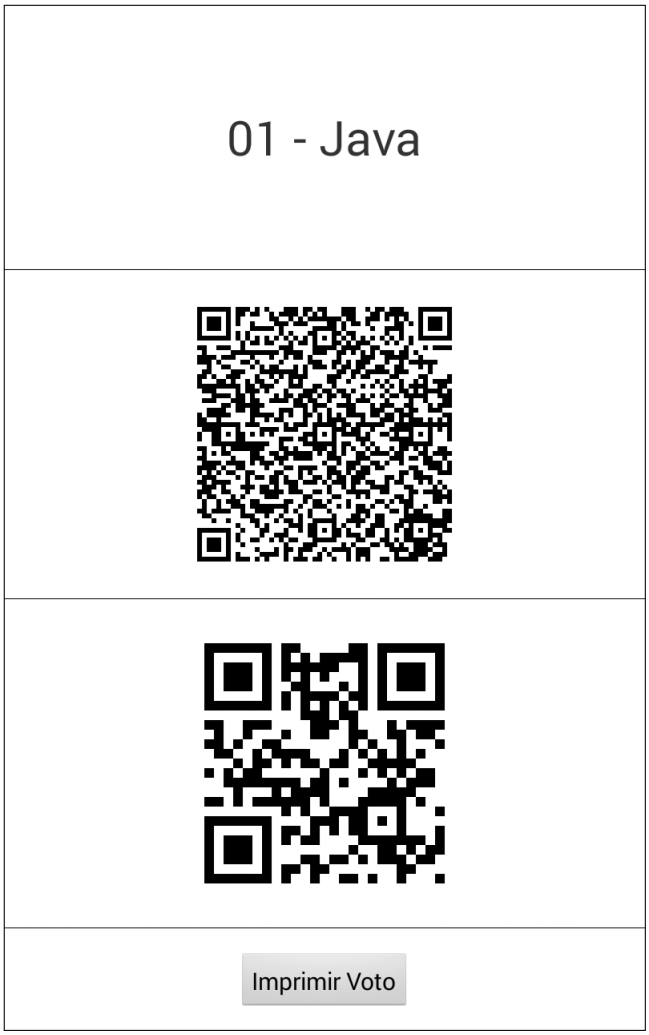

Figura 4.3: Papeleta mostrada en Tablet

Posteriormente, el votante debe apretar el botón "Imprimir Voto", con lo cual procede a abrir la ventana de impresión de *Android*. Reiterar que lo ideal hubiera sido que la impresora estuviera conectada de manera directa a la *tablet*, pero se utilizó una impresora conectada en red. Luego de esto, se imprime la papeleta tal cual apareció en la pantalla. Se muestra, para finalizar, un mensaje explicando que se debe proseguir a la siguiente fase de la votación, que es la verificación de la encriptación.

4.3. Segunda parte: Verificación de la Encriptación

Luego de terminar de utilizar la aplicación en la *tablet* (obteniendo la impresión de la papeleta) se procede a utilizar el verificador de la encriptación. Para esta implementación se utilizó una *Raspberry PI B+*[12], que es un micro-computador con recursos bien limitados, pero que cumplían lo necesario para realizar el trabajo. Además, para leer los códigos *QR* se utilizó un lector de códigos 2D *Motorola Symbol DS-4308*, el cual sirve para leer múltiples códigos, en particular, *QR*s. Se decidió ocupar un lector para aumentar la velocidad de lectura de los códigos, y así hacer más fácil la experiencia para los votantes.

Pensando en este hardware, se desarrolló una aplicación, *BallotVerification*[13], escrita en *Java*, que tendría como misión verificar la encriptación que se realizó. Para ello es necesario que la aplicación posea la clave pública *pk* utilizada para encriptar, además de la lista de los candidatos `candidates.xml`.

La aplicación le solicita al votante que escanee el primer código *QR*, es decir, la encriptación firmada. Luego de leer ese código, le solicita que escanee la segunda, es decir, la aleatoriedad. Con esos datos, la aplicación procesa el algoritmo 1 descrito en la parte 3.3 de este trabajo, encriptando las opciones utilizando la misma biblioteca de *Paillier* utilizada en la *tablet*. El resultado del algoritmo es mostrado al votante, es decir, la selección que está encriptada en el código. Si ésta es la misma que el votante quiso realizar, le es explicado que doble la primera parte de su papeleta y proceda a salir de la cabina y dirigirse a la mesa de votación. En cambio, si es que lo encriptado no es lo que el votante pretendía votar, o no se encuentra una opción válida encriptada, el votante puede volver a la *tablet* e imprimir una nueva papeleta. No hay problema con lo que haga el votante con la otra papeleta (la mala), debido a que ese voto nunca se emitirá.

La razón de utilizar una *Raspberry*, es sacarle el provecho a la modularidad del sistema, y debido a que el algoritmo de verificación es bastante simple de implementar y no necesita mayores recursos para computarlo. Así, la auditoría a realizarse a esta máquina se hace también más simple, ya que solamente necesita *Java* y un sistema operativo minimal, como lo es *Raspbian Wheezy*[14], una variación de *Debian* para *Raspberry*.

4.4. Tercera parte: Autentificación y Emisión del Voto

La siguiente aplicación dentro del proceso de votación, es la responsable de emitir el voto, denominada *CaptureQRBallot*[15]. Al igual que el resto, fue escrita en *Java*. Para leer los códigos *QR* fue utilizada una cámara para computador, en vez de un lector de códigos (se probó con una cámara para analizar alternativas). Se pensó utilizar también una *Raspberry* para correr esta aplicación, pero lamentablemente no funcionaba bien la cámara (faltaban controladores), por lo que finalmente se decidió en utilizar un *notebook* común y corriente.

[12] `http://www.raspberrypi.org/help/what-is-a-raspberry-pi/`
[13] `https://github.com/CamiloG/BallotVerification`
[14] `http://www.raspbian.org/`
[15] `https://github.com/CamiloG/CaptureQRBallot`

Esta aplicación va a ser utilizada solamente por los vocales de mesa, por lo que no se procuró mucho en poseer una interfaz muy llamativa (en la consideración que será usada por personas especializadas). La aplicación es bien simple, y utiliza un par de bibliotecas externas. Al igual que aplicaciones anteriores, utiliza *ZXing* para el manejo de los códigos *QR*. Para la captura con la cámara, utiliza *webcam-capture*[16]. Por último, para la verificación de la firma, hace uso nuevamente del paquete `java.security`.

De antemano se necesita poseer todas las claves públicas de los votantes para realizar la verificación de las firmas digitales. Al recibir un votante que sale con su papeleta de la cabina, el vocal de mesa debe destruir el segundo código *QR* (aleatoriedad), ya que el votante podría demostrarle a alguien como votó utilizando esta aleatoriedad. Luego de esto ejecuta la aplicación, que solicita la id (RUT) del votante para buscar su clave pública; posteriormente se abre la cámara para leer el primer código *QR* (encriptación y firma). Luego se hace la verificación de la firma utilizando los métodos del paquete `java.security`. Si se comprueba que la firma es válida, se procede a registrar el voto, el cual se guarda encriptado tal como viene, y se almacena localmente[17]. Si no fuera válida, ver sección 3.6.5, donde se discute el manejo de errores. Luego de esto, el votante separa la primera parte de la papeleta y la deposita en una urna. Con esto se termina el proceso de votación para una persona en particular. Todo esto se repite para cada persona que viene a votar durante todo el período que duren las votaciones.

4.4.1. Conteo de Votos

Lo último que hay que realizar para cerrar el proceso de votación, es el conteo de votos. Para esto se implementó una aplicación, *VotesCounting*[18], la cual tiene como objetivo recorrer los votos encriptados y desencriptarlos de manera parcial. Para esto es necesario tener de antemano la lista de todos los votos encriptados, además de cada una de las w partes de la llave privada, necesarias para realizar la desencriptación. Por simplicidad en esta implementación no se realizó una desencriptación distribuida, sino que se hace la desencriptación de manera central, juntando las w partes entregadas por cada una de las autoridades.

Para la desencriptación parcial, se hace uso de la misma biblioteca de *Paillier* utilizada anteriormente para encriptar. Al comenzar la aplicación, ésta calcula la multiplicación de todos los votos encriptados, y este valor se desencripta parcialmente con cada una de las w partes de la clave privada que se tienen (se realiza de manera centralizada, pero simulando la desencriptación parcial de cada autoridad). Luego, éstas desencriptaciones se unen y se despliega el resultado, que es la suma total de cada uno de los candidatos (propiedad homomórfica del esquema).

Esta aplicación es bastante simple, y no necesita mayores recursos para su desempeño, por lo que podría ejecutarse en una *Raspberry*; eso si, para la experimentación, se utilizó un *notebook* común.

[16] `https://github.com/sarxos/webcam-capture/`

[17] Se deja registro en un par de *pendrives*. Lo ideal debe ser almacenarlos además en un servidor público para posterior revisión de los votantes que su voto será/fue contabilizado.

[18] `https://github.com/CamiloG/VotesCounting`

4.5. Consideraciones generales de la implementación

4.5.1. Representación de los votos

Los votos, en todas las aplicaciones, son representadas por un `byte[]`, de largo igual al número de candidatos más 2 (una para tener el total de votos y la otra para los votos en blanco). Entonces un voto por el candidato i ($1 \leq i \leq n$) se representa como un `byte[] voto`, con 0s en todas las posiciones, excepto en 2: `voto[0]` y `voto[i]`.

Con esto, si sumamos todos los votos y dejamos el resultado en `byte[] total`, tendremos el total de votos emitidos en `total[0]`, además de la cantidad que posee cada candidato i en `total[i]`. Esto genera un problema, ya que se pueden tener a lo más 127 ($2^7 - 1$) votos, debido a que después se produce un *overflow* y se pierde la regularidad. Solucionar esto no resulta complicado, pero para esta implementación (por simplicidad) se prefirió hacer muestras menores a ese número.

Capítulo 5

Experimentación y Pruebas

5.1. Consideraciones preliminares de la experimentación

Para la experimentación del sistema se hicieron varias pruebas de usabilidad con estudiantes mismos del Departamento de Ciencias de la Computación de la Universidad de Chile. Si bien no representa un espectro muy variado de personas (son personas muy familiarizadas con la tecnología), ya que su adaptabilidad al sistema será mucho más rápida que si se considera una muestra más representativa de la sociedad. Los resultados se comentarán de manera general, y no se adentrará en detalles particulares de lo que opinó cada persona.

Además de las pruebas de usabilidad, se hicieron exhaustivas pruebas a que cada uno de los módulos implementados estén funcionando bien, lo que se podrían llamar "pruebas de concepto". Lo importante acá es que el sistema en completo esté funcionando, y eso no significa que los módulos por separado estén haciendo bien su trabajo, sino que se necesite que el *output* de uno, sea bien recibido como el *input* del siguiente. Obviamente se hicieron pruebas además que el resultado total entregado por la parte electrónica, concordara con el resultado final entregado por los votos depositados en la urna ($VVPAT$).

Cabe recalcar que las pruebas que se realizaron, no siguieron los métodos formales para testear la usabilidad de un sistema. Sino más bien fueron pruebas realizadas para ver el funcionamiento de la implementación realizada y descubrir problemas específicos del sistema. Por ello no se ahondará en especificar número de participantes y el tipo de pruebas realizadas. Es importante señalar también que los objetivos principales de este trabajo era el diseño del sistema, además de presentar una posible implementación. Las pruebas experimentales del sistema no se presentaron como un objetivo a cumplir.

5.2. Discusión de los resultados obtenidos

5.2.1. Pruebas de Usabilidad

Después de utilizar el sistema de votación por completo, los participantes contestaron una encuesta[1], donde se recolectaron las impresiones sobre la usabilidad y entendimiento del sistema. Las conclusiones más importantes fueron:

1. Explicar el porqué de cada uno de los pasos: la principal crítica viene dada por el desconocimiento que provocan los pasos más novedosos dentro del sistema, que son la firma digital, la verificación del voto y la destrucción de la aleatoriedad utilizada. Si bien los votantes pueden seguir el *workflow* del proceso de votación, este lo realizan en muchos ratos "a ciegas", bajando así su apreciación del sistema en general.
2. Buena apreciación de la primera parte: excepto por el hecho de firmar la selección, la primera parte del sistema fue la mejor recibida, principalmente por la razón de utilizar dispositivos conocidos y amenos, como es la *tablet* junto con la impresora. Ésta fue la parte que se le dio más importancia dentro de la actual implementación, debido a que es la principal interacción entre el usuario y el sistema.
3. Difícil adaptación al sistema: utilizar el sistema por primera vez, viniendo de un estilo clásico de votación, resulta muy engorroso. Esto acompañado al hecho que el sistema parece ser muy largo, aunque solo se agrega un paso más (verificación) a un sistema clásico (selección y emisión); esto da la sensación que el proceso se extiende demasiado, ayudado por el hecho que es este paso justamente el más complicado de entender. Faltó, ya sea una "capacitación" previa al uso del sistema, o una mayor explicación de los pasos a seguir durante el proceso (esto será abordado con más detalles en el próximo Capítulo).
4. Buena percepción de seguridad: si bien la gran mayoría de los votantes no entendió algunas de las partes del sistema (lo cual es un problema a resolver), dio la sensación que el sistema sí brindaba seguridad al proceso de votación, es decir, que existe tanto privacidad del voto como integridad en el resultado final. Además se valora y se entiende rápidamente el porqué de la necesidad de un *VVPAT*, aunque este resulta extraño la primera vez, surgiendo la pregunta: ¿porque la necesidad de un voto en papel si lo estoy haciendo electrónicamente? Afortunadamente, al explicar las razones brindadas previamente en este trabajo, se valora y se comprende.

5.2.2. Pruebas de Concepto

Al realizar un sinnúmero de pruebas de concepto, los resultados más importantes fueron:

1. Refrescar pantalla de aplicación de verificación: en la aplicación de verificación, como no tiene teclado, solo utiliza como periférico el lector de códigos *QR*, queda el registro de la verificación anterior en pantalla, y es necesario borrarlo para que el próximo votante

[1] `http://http://goo.gl/forms/qOgUSWFsrV`

no pueda ver como votó el anterior a él.

2. Automatizar transmisión de datos entre aplicaciones: si bien en un proceso de votación real hay más tiempo para realizar procesos preliminares (semanas), es necesario automatizar el proceso de envío de datos de una aplicación a otra. Entre estos datos que comparten 2 o más aplicaciones están la lista de candidatos, claves públicas (tanto de los votantes como del esquema de encriptación) y encriptación de los votos. Esto se puede solucionar montando un servidor en línea que posea estos datos, y que las aplicaciones al momento de arrancar las descarguen de ahí. Esto es comentado en mayor detalle en el próximo Capítulo.

Capítulo 6

Trabajo Futuro

6.1. Aspectos a mejorar del diseño

6.1.1. Adaptación a votación por Internet

Si bien la votación por Internet ha tenido variadas críticas por parte de varios expertos en la materia[1], sobre todo para utilizarse en instancias importantes de decisión, como lo son votaciones de índole nacional, o incluso más locales, pero que representen una elección de índole político-social; hay instancias en que la votación por Internet no representa un mayor problema, sobre todo, debido a que la repercusión de dichas elecciones es "baja". Para esas instancias, sería bueno que el sistema pudiera adaptarse y permitir una versión para votar por Internet. Si bien se eliminan algunos aspectos importantes del sistema (*VVPAT*, por nombrar uno), puede que permitirse algunos "relajos" impliquen un mayor uso del sistema, teniendo más experimentaciones, necesarias para siempre seguir avanzando en ir mejorándolo día a día. Algunas cosas que no pueden eliminarse son la modularidad del sistema, además del esquema a utilizar, ya que brinda la posibilidad de la desencriptación parcial, elemento vital para asegurar una real privacidad del voto.

6.1.2. Firma realizada por el votante

Actualmente el sistema solicita la clave privada del votante para poder firmar el voto encriptado, haciendo que la máquina posea la clave privada de cada participante de la elección, pudiendo realizar una suplantación de identidad en un posible ataque interno. Esto se contrarresta actualmente haciendo que solo cuente el primer voto, ya que es necesario que la persona vote una vez para poder obtener la firma privada. Lo ideal sería que esta máquina nunca supiera la clave privada del votante, permitiéndole a él, a través de una aplicación propia que posea, firmar el voto encriptado, y devolver la firma misma, y no la clave. Esto ahorraría un posible ataque, haciendo al sistema más robusto, y dándole más seguridad al

[1] `https://www.usenix.org/conference/evtwote13/workshop-program/presentation/wagner`

votante, que nunca brinda su clave privada, y que la firma la realiza él, en su propio dispositivo. Obviamente la idea es brindar una aplicación que haga el trabajo, y el votante solo deba darle la clave privada a esta aplicación, y le permita firmar el voto (o posteriormente, cualquier documento) de manera separada de la máquina que encripta.

6.2. Aspectos a mejorar de la implementación

6.2.1. Montar un *bulletin board* en línea

En la actual implementación los votos eran respaldados solamente de manera local, en un par de *pendrives* (para poseer redundancia). Una mejora sustancial sería que esos votos encriptados sean subidos a un servidor público (*bulletin board*), y sean vistos por toda la gente, así los mismos votantes pueden corroborar que su voto va a ser contabilizado en el recuento final. Además esto le brinda mayor transparencia al proceso, ya que si existe alguna duda sobre el resultado, siempre se pueden volver a contabilizar, ya que los votos se encuentran públicos. Obviamente se mantienen encriptados, por lo que nadie puede saber como votó cada persona. Junto con esto, otra mejora que viene de la mano es realizar una aplicación que facilite la comprobación de que el voto será contabilizado, solamente escaneando el código *QR* que cada votante se queda (el voto encriptado y firmado), y que posteriormente busque ese voto dentro del *bulletin board* e informe al votante si aparece o no. Si no llegase a aparecer, y es un voto válido, se puede ir a informar a las autoridades para que expliquen porque ese voto válido no está siendo contabilizado, ya que es este mismo conjunto de votos que será posteriormente procesado para revelar el resultado final. En este caso, el voto también debiera ser firmado por la máquina donde se realiza la selección (cada máquina posee un par de claves que puedan ser verificadas posteriormente). Esto permitiría demostrar a un juez que el voto efectivamente se emitió por una máquina válida, y no fue creado de manera externa al sistema.

6.2.2. Descentralizar la desencriptación parcial

La desencriptación parcial se hace de manera centralizada, es decir, cada autoridad debe brindar su parte de la clave privada, para que una sola aplicación realice las desencriptaciones correspondientes a cada una. Esto puede ser mejorado a que cada autoridad no tenga que brindar su parte de la clave, sino que simplemente acceda a una aplicación, que descargue los votos encriptados del *bulletin board* (o que descargue solamente la multiplicación de estos, realizada en el mismo servidor), y luego los desencripte con su parte de la clave, devolviendo al servidor el resultado de dicho proceso. Esto permite que cada parte de la clave queda solamente en manos de la autoridad correspondiente, y ésta nunca se revela, haciendo imposible que se pueda utilizar para desencriptar los votos por separado (siendo éste un posible ataque interno hacia el sistema). Posteriormente en el mismo servidor se reciben cada una de las desencriptaciones (las cuales no revelan nada por separado), se unen y se revela el resultado, que viene siendo el resultado total de la elección.

6.2.3. Conexión directa Impresora-Tablet

Como fue comentado anteriormente, realizar esta conexión se volvió más complicado de lo que se pensaba en un principio, debido a que de manera nativa *Android* no provee este servicio, por lo que es necesario modificar el controlador de la impresora para que admita la conexión por cable USB directamente desde el dispositivo móvil. Esto es posible de realizar, ya que existe una aplicación llamado *StarPrint*[2], la cual realiza esta acción, pero lamentablemente para sacarle una marca de agua que trae es necesario pagar por ella. Por eso no se utilizó, pero es una buena medida para saber que esto se puede realizar de buena manera, e integrarlo a la aplicación sería una mejora sustancial a la seguridad del sistema en general, debido a que una conexión por red inalámbrica es insegura, debido a los sinnúmeros de ataques posibles de realizar, poniendo en jaque la privacidad del voto de cada uno de los votantes.

6.2.4. Eliminación de claves privadas

Luego de crear las claves (tanto las de la firma digital para cada votante como las necesarias para le encriptación y desencriptación de los votos), éstas se reparten a cada uno de los participantes (en un caso a los votantes y en el otro a las autoridades). Lamentablemente la implementación actual no asegura que éstas sean eliminadas posteriormente de ser entregadas a sus respectivos "dueños", por lo que podían filtrarse y tener problemas posteriores de suplantación de identidad o privacidad del voto. Es necesario implementar un mecanismo, en la aplicación misma, que asegure que dichas claves son olvidadas por el sistema al momento de ser entregadas.

6.2.5. Revisión rigurosa a paquetes y bibliotecas externas

Para esta primera versión se privilegió hacer funcionar el sistema, frente a asegurar una total (o casi total) protección de los datos que se van procesando a lo largo del sistema. Con esto, no se dice que lo que hay actualmente vulnere totalmente la seguridad, pero para asegurarlo fehacientemente es necesario realizar un análisis mucho más acabado y minucioso de todos los paquetes nativos de *Java* y *Android* utilizados, además de las bibliotecas externas que se utilizaron para esta implementación. Con esto, se podría asegurar un buen funcionamiento contra posibles ataques al sistema, dando fe que la seguridad de éste no será vulnerada, ya que todos los datos que se fueron manejando a lo largo de proceso están bien resguardados.

6.2.6. Mejoras de usabilidad

Todos los comentarios obtenidos a partir de las experimentaciones realizadas con el sistema deben ser analizados e implementados. Sobre todo hay dos cosas claves, vitales para una

[2] `https://play.google.com/store/apps/details?id=com.ivc.starprint`

buena comprensión (y posterior uso) del sistema: explicación de las razones de los distintos módulos, y una mejor guía de los pasos a seguir. Se pretende que el sistema sea *user-friendly*, sin la necesidad de una "capacitación" o explicación previa de los pasos a seguir. Esto implica un desafío con respecto a que el sistema sea intuitivo de utilizar (qué pasos seguir y cómo realizarlos). El porqué de su uso (implicancias de seguridad sobre todo) pueden ser explicados en el mismo momento en que son utilizados. Obviamente se necesita que sea explicativo para todos los votantes, con y sin conocimientos de computación o seguridad previos.

6.3. Próximos pasos y horizonte del proyecto

Con las mejoras a realizar nombradas en este capítulo, el sistema de votación quedaría bastante robusto frente a posibles ataques, ya sean estos externos o internos. Es por ello que se piensa seguir trabajando en mejorar el sistema, para posteriormente utilizarlo en instancias reales de votación, estableciendo un protocolo claro al momento de enfrentar algún problema, o surjan sospechas de que el sistema está funcionando mal.

Con eso en mente, se piensa "promocionar" el sistema, ofreciéndolo en variadas instancias donde se pueda utilizar, pensando sobre todo en votaciones de índole estudiantil, donde la implementación de un sistema electrónico es más sencillo, debido a la cercanía que tienen los jóvenes con la tecnología.

De esas instancias es necesario ir sacando experiencias, que ayuden al momento de implementar el sistema en circunstancias donde los votantes no estén tan cercanos a la tecnología, y el cambio deba hacerse más gradualmente, o pensar mejor en cómo enseñar a seguir el proceso de votación. Aunque es bueno recalcar que la idea es que un sistema de votación necesite la menor cantidad de explicación previa, debido a que es un derecho ganado por todos los ciudadanos, y no puede ser que alguien se quede afuera del proceso solamente por no entender el sistema. Este es un desafío grande que se enfrenta, y para el cual es necesario que se pasen por muchas fases de pruebas para identificar bien los pros y contras del sistema, arreglando las contras y fortaleciendo los pros.

Capítulo 7

Conclusiones

Uno de los mayores problemas de los variados sistemas electrónicos de votación que han fracasado en distintas partes del mundo es no tomar en cuenta como un pilar fundamental al usuario final. Y esto se percibe desde varias aristas:

- No aseguran que el usuario final pueda utilizar el sistema sin peligro de que se violen algunos derechos fundamentales a la hora de participar en una votación (privacidad, integridad, coerción, etc.).
- No le permiten al usuario que pueda realizar una auditoría al sistema, para transparentar los procesos que se realizan y dando a conocer cada línea de código que procesan los votos de cada uno.
- No hay mayor interés en explicarle al usuario cuáles son las bases que rigen que el sistema realmente no revelará los votos de cada persona, o no cambiará dichos votos en el intertanto.

Y así se pueden seguir enumerando problemas que surgen de no considerar al votante como una pieza clave del sistema. El éxito que llegue a tener el sistema depende principalmente de la confianza que inspire en la gente, sobre todo en el sentido de la privacidad que entregue a la selección que realice cada votante. Esto, sin dejar de lado la usabilidad que posea, y que no implique una dificultad mayor a la hora de efectuar los sufragios (comparado a un sistema clásico de votación).

Teniendo eso en cuenta, la propuesta de este proyecto apunta a considerar el usuario como una parte imprescindible del sistema, recayendo en él gran parte de la responsabilidad en el aseguramiento de que éste funcione de manera correcta (añadiendo la verificación por parte de él, de la encriptación realizada). Si bien esto suma problemas de usabilidad del mismo sistema, debido principalmente a lo inusual que resulta para los votantes este paso, eso no debería suponer "saltarse" este paso. El principal desafío es lograr que los votantes entiendan el porqué de la necesidad que el sistema tiene que sean partícipes de este proceso de verificación. Con eso superado, los mismos usuarios permiten un sacrificio en la usabilidad dentro del sistema (extender el tiempo necesario para poder emitir el voto) por un aseguramiento al final del día que los resultados obtenidos son los que deberían haber sido, con ellos como los principales actores dentro de esa capa de seguridad obtenida.

Si bien el sistema, como está actualmente implementado, posee varios problemas de usabilidad y capacidad de poder expresar a los usuarios las razones de cada uno de los pasos, se puede observar que las mejoras que se pueden realizar (nombradas en el Capítulo anterior) lo dejaría como un sistema candidato para poder utilizarlo en instancias más formales de elección, debido a que sus distintas partes permiten asegurar la integridad, privacidad del voto, autentificación, accesibilidad, prevención de coerción, transparencia, entre otras características esenciales de un sistema de votación.

La principal conclusión que deja este trabajo es la necesidad que poseen los mismos sistemas de votación electrónica que sus usuarios se hagan partícipes del proceso de votación, siendo ellos los principales actores a la hora de asegurar requisitos de seguridad imprescindibles que debiese poseer. El éxito de un sistema de votación (o cualquier sistema de software) depende de la percepción que posean los usuarios finales del sistema sobre que tan útil les resulta. Si un sistema de votación fracasa en ese aspecto, se generan mayores problemas que si le sucediera a un sistema de software en general, produciéndose sospechas a que el proceso está viciado, además de desincentivar a los votantes a que participen del proceso eleccionario. La votación electrónica debe verse como un problema multidisciplinario, donde la Computación debe saber lidiar con el hecho que el software a elaborar está siendo utilizado para una de las etapas más críticas del acontecer ciudadano y político de un país (o cualquier localidad u organización). Es tiempo de darle la importancia que se merece, y empezar a idear soluciones acordes al contexto de cada situación, poniendo como eje principal, al usuario final.

Capítulo 8

Bibliografía

[1] Ben Adida. Helios: Web-based open-audit voting. In *USENIX Security Symposium*, volume 17, pages 335–348, 2008.

[2] Andrew W Appel. Security seals on voting machines: A case study. *ACM Transactions on Information and System Security (TISSEC)*, 14(2):18, 2011.

[3] Diego F. Aranha, Marcelo Monte Karam, Andre de Miranda, and Felipe Scarel. Software vulnerabilities in the brazilian voting machine. 2013.

[4] Patrick Baxter, Anne Edmundson, Keishla Ortiz, Ana Maria Quevedo, Samuel Rodrıguez, Cynthia Sturton, and David Wagner. Automated analysis of election audit logs. *Proceedings, EVT/WOTE*, 2012.

[5] Jonathan Ben-Nun, Niko Fahri, Morgan Llewellyn, Ben Riva, Alon Rosen, Amnon Ta-Shma, and Douglas Wikström. A new implementation of a dual (paper and cryptographic) voting system. In *Electronic Voting*, pages 315–329, 2012.

[6] Joseph A Calandrino, Ariel J Feldman, J Alex Halderman, David Wagner, Harlan Yu, and William P Zeller. *Source code review of the Diebold voting system.* California Secretary of State, 2007.

[7] Stephen Checkoway, Ariel J Feldman, Brian Kantor, J Alex Halderman, Edward W Felten, and Hovav Shacham. Can dres provide long-lasting security? the case of return-oriented programming and the avc advantage. *Proceedings of EVT/WOTE*, 2009, 2009.

[8] Danny De Cock and Bart Preneel. Electronic voting in belgium: Past and future. In *VOTE-ID*, pages 76–87, 2007.

[9] Frederick G Conrad, Benjamin B Bederson, Brian Lewis, Emilia Peytcheva, Michael W Traugott, Michael J Hanmer, Paul S Herrnson, and Richard G Niemi. Electronic voting eliminates hanging chads but introduces new usability challenges. *International Journal of Human-Computer Studies*, 67(1):111–124, 2009.

[10] Rop Gonggrijp, Willem-Jan Hengeveld, Andreas Bogk, Dirk Engling, Hannes Mehnert, Frank Rieger, Pascal Scheffers, and Barry Wels. Nedap/groenendaal es3b voting computer: a security analysis. *Available via the" We don't trust voting computers" website at http://wijvertrouwenstemcomputersniet. nl/Nedap-en*, 2006.

[11] Carlos Vegas González. The new belgian e-voting system. *Unpublished paper*, 2012.

[12] Gurchetan S Grewal, Mark D Ryan, Sergiu Bursuc, and Peter YA Ryan. Caveat coercitor: Coercion-evidence in electronic voting. In *Security and Privacy (SP), 2013 IEEE Symposium on*, pages 367–381. IEEE, 2013.

[13] J Alex Halderman, Eric Rescorla, Hovav Shacham, and David Wagner. You go to elections with the voting system you have: Stop-gap mitigations for deployed voting systems. *EVT*, 8:4–4, 2008.

[14] Joseph Lorenzo Hall. Transparency and access to source code in e-voting. In *USENIX/ACCURATE Electronic Voting Technology Workshop*, 2006.

[15] Paul S Herrnson, Richard G Niemi, Michael J Hanmer, Benjamin B Bederson, Frederick G Conrad, and Michael Traugott. The importance of usability testing of voting systems. In *Proceedings of the 2006 USENIX/ACCURATE Electronic Voting Technology Workshop*, 2006.

[16] Mads Jurik and Jesper Buus Nielsen. A generalization of paillier's public-key system with applications to electronic voting. In *PYA Ryan*. Citeseer, 2003.

[17] Tadayoshi Kohno, Adam Stubblefield, Aviel D. Rubin, and Dan S. Wallach. Analysis of an electronic voting system. In *IEEE Symposium on Security and Privacy*, pages 27–, 2004.

[18] Leontine Loeber. E-voting in the netherlands; from general acceptance to general doubt in two years. In *Electronic Voting*, pages 21–30, 2008.

[19] Epp Maaten and Thad E. Hall. Improving the transparency of remote e-voting: The estonian experience. In *Electronic Voting*, pages 31–46, 2008.

[20] Rebecca Mercuri. Physical verifiability of computer systems. In *5th International Computer Virus and Security Conference*. Citeseer, 1992.

[21] Pascal Paillier. Public-key cryptosystems based on composite degree residuosity classes. In *Advances in cryptology—EUROCRYPT'99*, pages 223–238. Springer, 1999.

[22] Rodrigo Porras. Extensión y mejora de un sistema de votación electrónica para hacerlo más robusto, universalmente verificable, fácilmente usable y práctico, 2012.

[23] Ronald L Rivest, Adi Shamir, and Len Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*, 21(2):120–126, 1978.

[24] Adi Shamir. How to share a secret. *Communications of the ACM*, 22(11):612–613, 1979.

[25] Scott Wolchok, Eric Wustrow, J. Alex Halderman, Hari K. Prasad, Arun Kankipati, Sai Krishna Sakhamuri, Vasavya Yagati, and Rop Gonggrijp. Security analysis of india's electronic voting machines. In *ACM Conference on Computer and Communications Security*, pages 1–14, 2010.

[26] Scott Wolchok, Eric Wustrow, Dawn Isabel, and J. Alex Halderman. Attacking the washington, d.c. internet voting system. In *Financial Cryptography*, pages 114–128, 2012.