VNIVERSIDAD NACIONAL AVPNOMA DE MEXICO

UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

POSGRADO EN CIENCIA E INGENIERÍA DE LA COMPUTACIÓN

IMPLEMENTACIÓN DE UN SISTEMA DE VOTACIONES EN INTERNET UTILIZANDO UN PROTOCOLO BASADO EN CRIPTOGRAFÍA DE CURVAS ELÍPTICAS

T E S I S

QUE PARA OBTENER EL GRADO DE:

MAESTRO EN INGENIERÍA (COMPUTACIÓN)

P R E S E N T A:

OTONIEL MANUEL ORTIZ RUIZ

DIRECTOR DE TESIS: DR. GERARDO VEGA HERNÁNDEZ

México, D.F. 2005.

Dedico este trabajo a quienes desde el principio y hasta el final me brindaron su incondiconal apoyo.

Para y Por Mis Padres.

Agradezco a mi asesor el Dr. Gerardo Vega Hernández por su colaboración y paciencia, sin su ayuda esta Tesis no hubiera podido realizarse.

A su vez quiero reconocerle a mis otros sinodales:

Dr. Enrique Daltabuit Godas
Dr. Francisco Garcia Ugalde
Dr. Héctor Benítez Pérez
M. en C. Gustavo Arturo Marquez Flores

Por la atención y el valioso tiempo que dedicaron a este trabajo

Mi gratitud también queda con Lourdes González y Amalia Arriaga, sus consejos en diversos momentos fueron fundamentales.

Gracias por su constante aliento y su valioso apoyo:

A mis hermanos, Miguel y Rocio.

A mis otros hermanos, Fernando, Francisco y Horacio.

Y a todas aquellas personas que no menciono, pero que de diferentes formas contribuyeron para que pudiera culminar con esta difícil empresa.

Índice General

Introducción

Las votaciones pueden entenderse como una serie de elementos y métodos que permiten a cada uno de los miembros de un grupo seleccionar una opción entre varias, posibilitando a través de un conteo que la preferencia individual de la mayoría sea la que establezca la opción que todo el grupo dará por elegida. Este proceso usado a través del tiempo por diferentes grupos sociales en los que existen intereses divididos, se ve incrementado en importancia y complejidad de manera notable no sólo por el tamaño del conjunto de personas que se ven involucradas, sino además con la relevancia de la decisión que dicho grupo debe tomar.

Por ejemplo, en los países democráticos donde existe el principio de que los ciudadanos participen en las decisiones que conciernen a su nación, las votaciones cobran una importancia fundamental. Esto es debido a que el principal indicador por el cual se identifica la participación de la población en el destino de su país es por su capacidad para elegir a sus gobernantes; ésta decisión trascendente se lleva a cabo por medio del ejercicio del voto individual de cada elector. Si bien es cierto que la población cuenta con otros mecanismos válidos para influir en el destino de su país, estos carecen del gran impacto e importancia con el que las elecciones de gobernantes cuentan.

Sin embargo, a pesar de los avances tecnológicos, y de la importancia que las votaciones han tenido en el acontecer humano, los procesos electorales han permanecido sin cambios estructurales en sus mecanismos durante casi un siglo. Fue hasta las últimas dos décadas, en las que organizaciones públicas y privadas de todo el mundo han promovido el uso de sistemas de votaciones basadas en tecnologías alternativas como son votaciones por medio de máquinas mecánicas, tarjetas perforadas, lectores de hojas ópticas, urnas electrónicas, entre otras.

Actualmente el sistema tradicional de votaciones no ha sido desplazado por ninguno de los sistemas emergentes de su rol principal, pero cada vez se hace más evidente la tendencia a reemplazarlo.

Tan sólo en el año anterior a la publicación de esta Tesis durante las elecciones federales en Estados Unidos de Norteamérica, se hizo un extenso uso de urnas electrónicas, y se estuvo cerca de utilizar un sistema de Votaciones por Internet fuertemente impulsado por el pentágono. Sin embargo, a pesar de ello y de que varios sistemas de votaciones electrónicos se estén utilizando ampliamente en todo el mundo, estas tecnologías aún están lejos de ser maduras.

Al respecto, David Hill y Aviel Rubin, investigadores reconocidos en el campo de las votaciones electrónicas, declaran en el artículo *E-Voting Security* (ver Hill[25]): "Desde la perspectiva de la seguridad en cómputo, las votaciones electrónicas representan el peor de los mundos. Los activos en riesgo son extremadamente valiosos; las motivaciones de las personas que desean corromper los resultados son extremadamente poderosas; y el nivel de

sofisticación para que un atacante logre romper el sistema es extremadamente alto. Como si esto no fuera suficiente, el sistema debe descartar información que normalmente sería considerada crítica para la realización de auditorias: la relación de votantes y sus votos".

Dichas aseveraciones, pueden evidenciar de forma rápida la gran cantidad de factores que se deben tomar en cuenta durante el diseño de un sistema de votaciones; sin embargo, en la práctica no se ha puesto el cuidado adecuado ni en el diseño de los sistemas de votaciones ni en el empleo de estos por parte de las autoridades electorales implicadas. Ello se ha podido observar en varios países donde han surgido empresas privadas que han aprovechado la tendencia mundial de utilizar sistemas de votaciones alternos, para construir y vender urnas electrónicas que permitan realizar votaciones sin que se hayan realizado estudios a fondo en la seguridad que dichos productos proporcionan.

A pesar de que los resultados a estas situaciones no se han hecho esperar, y existen varias irregularidades en los procesos de votaciones asociados a las urnas electrónicas, la experiencia ganada en votaciones reales ha sido valiosa, se han creado recomendaciones que los diseñadores de estos sistemas han acatado y se observa una tendencia a querer establecer un estándar industrial que brinda certidumbre y sirva de marco de referencia tanto a diseñadores de sistemas de votaciones como a las autoridades electorales sobre la seguridad de dichos sistemas.

Los efectos de la adopción de otros sistemas de votaciones en el mundo no se han hecho esperar en nuestro país, y a través de congresos, simposios, pruebas piloto con urnas electrónicas brasileñas, etc., (ver [26]) se ha manifestado el interés de impulsar estas tecnologías para su uso. De ello surge la necesidad de que a nivel académico se estudie y se experimente con este tipo de tecnologías a fin de que no sea de la experiencia obtenida en prácticas reales de donde se obtengan elementos para reforzar la seguridad en los sistemas de votaciones.

El objetivo de esta tesis es diseñar y desarrollar un sistema de votaciones en Internet que integre criptografía fuerte, protocolos de votaciones robustos y la experiencia obtenida de los sistemas de votaciones que se han desarrollado a fin de crear un sistema de Votaciones en Internet, completamente funcional, de interfaz intuitiva y difícil de vulnerar.

A su vez se mantuvo en mente el objetivo de crear un diseño flexible que permitiera en lo más posible la portabilidad del sistema a diferentes plataformas sin cambios de fondo en su estructura.

También se busca que este trabajo sirva como base para futuros desarrollos de nuevos sistemas de votaciones que busquen maneras alternas de mejorar aspectos que no hayan sido cubiertos en este trabajo.

Como resultado de esta investigación, se realizó la implementación de un sistema de votaciones; ello implicó el diseño del protocolo de votaciones a utilizarse y el modelado de las bases de datos de las que haría uso, así como la programación de las entidades que lo integrarían.

La seguridad del sistema fue un aspecto que se cuido desde su fase de diseño; en él se buscó que este sistema pudiera soportar ataques externos, pero además que tuviera resistencia a aquellos ataques en los que se veían involucradas las autoridades responsables de realizar las votaciones.

El trabajo se encuentra organizado de en 5 capítulos:

Capítulo 1. Sistemas de Votaciones

Capítulo 2. Diseño del Sistema de Votaciones

Capítulo 3. Implementación del Sistema de Votaciones

Capítulo 4. Limitaciones, Amenazas y Escenarios de Implantación

Capítulo 5. Conclusiones

En el capítulo 1 se definen las características fundamentales de los sistemas de votaciones y se analizan los diferentes sistemas de votaciones utilizados en la actualidad. En el capítulo 2 se describe a detalle el protocolo su fundamento criptográfico y las entidades abstractas que se implementaron para construir el sistema de votaciones; a su vez se analizan y justifican las tecnologías empleadas para el desarrollo del sistema. En el capítulo 3 se describe cada uno de los componentes implementados en el Sistema de Votaciones. En el capítulo 4 se establecen las limitaciones y las amenazas a los sistemas de Votaciones en Internet y el como por medio diferentes escenarios de implantación, se busca mitigar o solucionar esos problemas. Finalmente en el capítulo 5 se plantean consideraciones finales sobre la seguridad del sistema así como futuras mejoras al sistema propuesto.

Capítulo 1 Sistemas de Votaciones.

En este capítulo, se puntualizarán los requerimientos que un sistema de votaciones debe cumplir y se dará una breve descripción de los sistemas de Votaciones más ampliamente utilizados. Lo anterior será buscando explicar de que forma estos sistemas buscan cumplir con las características que deben tener los sistemas de votaciones, además de mencionar las principales debilidades o inconvenientes que pudieran tener.

Cabe señalar que se dará énfasis en la descripción de las urnas electrónicas y de los Sistema de Votaciones por Internet, el primero por el impulso que se les ha dado en los últimos cinco años, y los segundos por tratarse del mismo tipo de sistemas de votaciones del que trata esta tesis.

Los tipos de sistemas de votaciones que actualmente se utilizan varían enormemente en la tecnología que usan. Por ello es posible encontrar en un mismo país que el sufragio se ejercite por medio de métodos tan variados que van desde votar por medio del correo postal, hasta realizarlo a través de sistemas electrónicos construidos exclusivamente para realizar votaciones.

A su vez los procedimientos que debe realizar el elector para ejercer su voto, variarán incluso cuando se utilizan los mismos sistemas, esto ocurre frecuentemente cuando el sistema del cual se está haciendo uso no es un sistema de Votaciones por si mismo. Sin embargo, para que las elecciones tengan validez, se parte del supuesto de que todos los sistemas de votaciones que se utilicen deben de cumplir con una serie de requisitos que a continuación se detallarán.

1.1 Requerimientos que debe cumplir un Sistema de Votaciones

Los requisitos de los sistemas de votaciones que podemos encontrar en Jefferson [28] y en Cranor [9], se pueden agrupar en requisitos obligatorios y deseables. Los primeros, se refieren a características forzosas de encontrarse en un Sistema de Votaciones; y los segundos a características que sin ser indispensables, le pueden dar un valioso agregado al sistema.

1.1.1 Obligatorios

Los siguientes aspectos deben cumplirse en cualquier sistema de votaciones, sea manual o electrónico para que éste sea funcional. Esto es debido a que son precisamente estas características las que le permiten a un sistema de votaciones cumplir con la normatividad básica que se exige en una elección.

I Confiable

Esta es la característica más importante de cualquier sistema de votaciones, se refiere a los requerimientos que los votantes deben cumplir para poder ejercer su voto y como los votos son manejados cuando estos han sido emitidos

Si un sistema no es confiable, el resultado final de la elección no será válido; siendo irrelevante cualquier otra característica que el sistema puede ofrecer

Un sistema de votaciones será Confiable si cumple con las siguientes características:

a) Exacto

- Todos los votos son almacenados de una manera segura, de tal forma que ninguno debe perderse.
- No es posible alterar un voto.
- Todos los votos válidos y sólo los votos válidos serán contados

b) Legal
- El sistema Identifica a los votantes válidos
- Solo votantes válidos pueden ejercer su voto
- Los votantes válidos votan una sola vez.

II Confidencial

Dado que es prioritario que los votantes voten en completa libertad; el sistema de votaciones debe conservar la privacidad de los votantes. Se considera a un sistema de votaciones confidencial si cumple con el siguiente principio:

No hay manera de vincular al votante con su voto.

1.1.2 Requerimientos deseables.

Aunque si un Sistema de Votaciones es confiable y confidencial cuenta con los suficientes elementos para funcionar adecuadamente, existen otras características que diferencian a los sistemas de votaciones, estos pueden ser el factor decisivo en el momento de decidir que sistema es el más idóneo a utilizar. Los siguientes aspectos suelen ser encontrados en Sistema de Votaciones electrónicos.

Verificable

Un sistema puede ser verificable de forma completa o parcial.

La verificabilidad parcial permite a cualquier votante verificar que su voto ha sido contado de forma correcta.

La verificabilidad completa permitiría revisar que todos los votos han sido contados.

Practicidad

Un sistema es práctico si existen mecanismos para permitir a los votantes ejercer su voto con un mínimo de acciones requeridas por ellos y sin la necesidad de que los electores tengan un entrenamiento especializado o bien necesiten contar con equipo complejo para el ejercicio de su voto.

Accesibilidad

Podemos entender por grado de accesibilidad de un sistema de votaciones, a la facilidad de los votantes para estar en el lugar físico desde donde puedan votar.

Velocidad

El conteo de los votos debe de tomar el menor tiempo posible.

1.2 Sistemas de Votaciones

Existen diferentes tecnologías alternas alrededor del mundo para realizar votaciones; entre ellas podemos encontrar sistemas muy variados como el voto por teléfono, por correo postal, y diversos sistemas electrónicos; sin embargo, para efecto de este trabajo, únicamente se analizarán aquellos que son más ampliamente utilizados.

1.2.1 Antecedentes de los Sistemas de Votaciones.

Durante la mayor parte del siglo XIX, un votante típicamente tomaba un trozo de papel en donde estaban listados los nombres de los candidatos y marcaba su selección a la vista de todos.

Procedía entonces a jurar ante un juez, con la mano en la Biblia, que no había emitido su voto con anterioridad. No existía ningún sistema de registro de los votantes, de modo que el juramento y el hecho de que posiblemente el juez reconocería al votante que intentará

emitir su papeleta por segunda vez, era lo que prevenía múltiples votos por parte de un solo votante.

Como es evidente, esta práctica tenía los siguientes inconvenientes:

Desventajas

No existía el derecho a la privacidad del voto, y era práctica legal y común que se hiciera campaña en el lugar en donde se efectuaban las elecciones.

El número y la secuencia de las papeletas no estaban controlados, de tal suerte que era difícil determinar si se habían agregado votos espurios a una determinada urna. Cualquier persona presente en el lugar de la votación podía determinar por quién había sido emitido un voto en particular; la compra y coerción de los votos era por lo tanto relativamente fácil.

En la actualidad, los sistemas de voto si bien es cierto no han logrado garantizar una fiabilidad completa en la confiabilidad y confidencialidad; al menos si buscan salvaguardar estos aspectos.

1.2.2 Votaciones con Boletas

Este sistema de votación es conocido también como "Boletas Australianas", ya que es el gobierno Australiano el primero en adoptar este sistema. Después de que en 1880 se dieron una serie de escándalos que involucraban la compra del voto (ver Fischer[16]) ; en 1889 Nueva Cork adopta el sistema de voto con boletas de papel.

Siendo uno de los métodos de votaciones más viejos, tiene un grado de confiabilidad y confidencialidad superior al de otras tecnologías. En nuestro país actualmente sigue siendo la única forma de voto válida. En algunos países como los Estados Unidos de Norteamérica es utilizado sólo por el 1% de los votantes principalmente como alternativa al voto en sitios rurales.

Características.

Los sistemas de voto que utilizan boletas de papel, se caracterizan por emplear boletas oficiales, impresas en papel que cuenta con los mecanismos de seguridad necesarios para evitar la emisión de boletas espurias. En estas boletas se imprime una lista con los nombres de todos los candidatos.

Las boletas de papel son idénticas, a excepción de un número serial consecutivo que generalmente se imprime para facilitar el control de las boletas, ayudando además a evitar sustitución de las boletas e introducción de boletas falsas en la urna.

Los votantes emiten su voto en privado, marcando las casillas que aparecen impresas junto al nombre del candidato de su preferencia. Finalmente, la boleta con el voto es depositada en una urna sellada.

Ventajas

Dado que las boletas son idénticas y además son llenadas en secreto, se hace muy difícil determinar con exactitud la elección de un votante en particular.

Desventajas

A pesar de las ventajas mencionadas previamente, se sigue dependiendo de la honestidad de los individuos involucrados en el proceso de votaciones; por ello la confiabilidad de éste sistema se sostiene en los siguientes supuestos:

- No existen cámaras escondidas.
- No hay sensores ocultos bajo la mesa en la que se marca la boleta.
- No hay análisis de huellas digitales en las boletas después de la elección.

Además, este sistema de votación, no elimina la posibilidad de alterar las boletas, ya que éstas pueden ser removidas de la urna, invalidadas o destruidas por funcionarios de casilla corruptos.

Dadas las condiciones necesarias, existe la posibilidad de generar boletas espurias.

No se elimina totalmente la posibilidad de compra y coerción del voto; un votante puede marcar la boleta de cierta forma que sea reconocible a quien ejerció la coerción.

1.2.3 Votaciones con Tarjetas Perforables

El sistema de votaciones de tarjetas perforadas fue inspirado en la máquina que el estadístico Herman Hollerith, construyó y utilizó con éxito en el censo de E.U.A. de 1890.

Este sistema de votaciones surgió en 1960 retomando la esencia del mecanismo de la máquina de Hollerith por dos profesores de Berkley que buscaban crear una máquina que permitiera hacer votaciones con tarjetas perforables a fin de obtener conteos de votos de forma veloz. Para una referencia completa del tema ver Joseph [30]

Aún cuando esta es una tecnología muy vieja que está por salir de uso, fue utilizada de manera muy extensa durante las últimas cuatro décadas en los Estados Unidos.

Características.

Los sistemas de votaciones de tarjetas perforables, son equipos mecánicos en el que emitir un voto consiste en introducir en el dispositivo de votaciones una gruesa tarjeta que vendría a ser la analogía de una boleta y perforarla. La perforación de cada tarjeta es interpretada por un equipo de conteo que puede ser una computadora auxiliada por un lector de tarjetas perforadas o bien un equipo electromecánico especializado en leer este tipo de tarjetas.

Los equipos para votar distan de ser complejos, pues no es más que un tablero que sirve de interfaz al votante para que este pueda ubicar fácilmente dónde tiene que realizar la perforación de la tarjeta a fin de ejercer su voto por el candidato de su preferencia.

Existen dos principales variantes en este sistema por el tipo de tarjetas que se utilizan. Los hay con boletas genéricas en las que sólo viene numerado el voto, (VotoMatic), y aquellas en las que las tarjetas tienen impreso el nombre del candidato (VotoData); obviamente éstas últimas se diseñan e imprimen exclusivamente para cada elección, por lo que implican una inversión superior, pero en cambio le dan cierto grado de seguridad al elector al permitirle verificar que la perforación en la tarjeta se hizo de forma correcta.

Ventajas

La principal ventaja que tiene a su favor este sistema de votaciones es el de poder obtener un rápido conteo automatizado de los votos a un costo en ocasiones quince veces menor que con otros sistemas automatizados (ver Fessenden Ford [18]).

Este sistema proporciona una evidencia física de la intención del voto de los electores.

Desventajas

Las votaciones con tarjetas perforadas son el sistema de votaciones que menos exactitud ha mostrado en comparación con los demás sistemas. Aún cuando su margen de error es mínimo, dos conteos difícilmente devuelven el mismo resultado; ello se debe a que en ocasiones los votantes no perforan completamente la tarjeta, volviéndola así una lectura ambigua para el dispositivo de votaciones.

Es posible alterar el dispositivo de votaciones para que tras insertar en él la tarjeta, ésta no embone correctamente y la marca se realice en un sitio que representará un voto para un candidato diferente al que el elector había elegido.

1.2.4 Votaciones con Lectores Ópticos

Es un sistema similar al de tarjetas perforadas, en éste el votante obtiene de las autoridades electorales una boleta con opciones que deberá marcar de acuerdo al candidato por quien desea votar. Terminada la votación, un lector óptico especializado procesa las boletas marcadas y realiza un conteo preciso de los votos.

Ventajas

Para muchos electores el proceso de llenar formas para lectores ópticos, es algo con lo que esta familiarizados. Esto se debe a que muchas encuestas, sorteos, censos y exámenes hacen uso de esta tecnología.

Rapidez y mayor exactitud que los equipos de tarjetas perforadas.

La forma que el elector marca para su procesamiento posterior en el lector óptico es una evidencia física del voto que ejerció, por lo que en caso de controversia puede utilizarse ésta para realizar un conteo manual.

Desventajas

Es común que los electores no marquen correctamente las opciones, provocando que el lector óptico no lea adecuadamente su voto.

1.2.5 Votaciones con Urnas Electrónicas

Características

Una urna electrónica, es un dispositivo electrónico, que permite a los electores ejercer su voto, almacenarlo, y realizar el conteo de todos los votos emitidos en ella de forma inmediata. Estos sistemas almacenan el voto como un registro electrónico que es posteriormente procesado para obtener el resultado de las elecciones. Comúnmente a este sistema que conocemos por urna electrónica se le conoce por DRE (Direct Register Electronic).

Estos dispositivos son utilizados bajo una logística similar al que el sistema de votaciones tradicional utiliza. Primeramente el ciudadano se identifica como un elector válido, y sólo entonces el DRE es habilitado para poder ejercer un voto.

Físicamente, los sistemas DRE tienen la apariencia ya sea de un cajero automático o de una computadora portátil (laptop). Por lo general llevan incorporadas tecnologías de asistencia para las personas con habilidades especiales, permitiéndoles votar sin necesidad de involucrar a otra persona.

En general proporcionan la gran conveniencia de llevar todo el proceso de votaciones de forma automática, ya que por medio de la urna electrónica el votante ejerce su voto, ésta lo almacena y finalmente realiza el conteo pertinente para proporcionar el resultado de la elección.

Elementos de una urna electrónica

Interfaz:
Es el medio de comunicación por el cual los votantes interactúan con la urna electrónica, esta puede consistir en botones o una pantalla sensible al tacto. Idealmente esta debe proporcionar un medio simple e intuitivo para que los electores puedan ejercer su voto.

Medio de Almacenamiento:
Es el dispositivo en el cual los votos son almacenados por la urna electrónica; idealmente debe cumplir con ciertos estándares de calidad a fin de asegurar que podrá almacenar de forma confiable e integra cada uno de los votos que los electores ejerzan en ella.

Sistema de Conteo:
Este sistema interno con el que cuentan todas las urnas electrónicas, es el encargado de realizar el escrutinio de los votos que almacenó durante la votación.

Tipos

Urnas con interfaz gráfica.

Este tipo de urnas cuenta con un monitor o una pantalla sensible al tacto en la que se asocia una imagen a los candidatos de tal forma que el votante pueda elegir pulsar entre una de ellas para emitir su voto.

Su practicidad es superior a cualquier otra, volviendo extremadamente sencilla su operación dado que el votante todo lo que tiene que hacer es pulsar sobre una pantalla sensible para ejercer su voto. Además existen modelos que cuentan con un sistema braille y dispositivos de audio que permiten el voto de invidentes.

Ventajas

- Gran Accesibilidad
- Facilidad de operación

Desventajas

No es viable realizar una auditoria eficiente sobre la confiabilidad del software que una urna con interfaz gráfica utiliza. Además de que en la mayoría de las veces, el código fuente es propiedad de la empresa que lo vende y el acceso a él está restringido; las rutinas para el despliegue gráfico, el manejo de los eventos del touchscreen, y el audio si es que se utiliza, incrementan notablemente la complejidad del código de la urna electrónica. Bajo dichas condiciones, a un programador malicioso le resultaría sencillo esconder un *backdoor* en alguna imagen, archivo de audio o en el programa mismo, mientras que la tarea de detectarlo tendría gran dificultad.

Éste tipo de urna es la más costosa de todas.

Urnas electrónicas sin interfaz gráfica

Generalmente el registro del voto lo emite un elector presionando un botón asociado a un candidato; el dispositivo almacena el voto y es capaz de realizar posteriormente el escrutinio de los votos. Este tipo de urnas al ser más simples son más susceptibles de ser auditadas en sus procedimientos internos.

Ventajas

Susceptibles de ser auditadas en su funcionamiento interno.

Desventajas

Más difíciles de operar, de accesibilidad limitada.

Urnas electrónicas con comprobantes de votos

Es una urna que puede o no tener interfaz gráfica, la cual imprime los votos que va almacenando y de esta forma proporciona un mecanismo eficaz para realizar una auditoria de los votos ejercidos y de su escrutinio.

Ventajas

Proporciona un mecanismo para comprobar la validez de la votación.

Desventajas

Mayor costo; introduce al paradigma de urna electrónica un problema más a ser analizado; el del esquema bajo el cual se imprimirán votos, para evitar su falsificación.

Experiencia de Urnas Electrónicas en Estados Unidos de Norteamérica.

A raíz de incidentes ocurridos durante las elecciones presidenciales del 2000, se perdió la confianza en los sistemas de votaciones más utilizados, los de tarjetas perforadas. Como respuesta el presidente que resultó electo George W. Bush creó la ley HAVA (Help America Vote Act), en la que se asignaron recursos para sustituir aquellos sistemas por sistemas de votaciones basados en urnas electrónicas.

Parte del proceso de la renovación de los sistemas de votaciones fue la creación de organismos encargados de crear estándares que permitieran que las Urnas Electrónicas, pudieran ser sistemas confiables; sin embargo, como podremos ver a continuación aún se está lejos de dicha meta.

A continuación se presenta un listado de las irregularidades que las urnas electrónicas han presentado en elecciones reales. Este listado no pretende ser una guía exhaustiva de las fallas que han ocurrido con urnas electrónicas, ni tampoco es un intento por descalificar a dicha tecnología; más bien se busca evidenciar el como se han explotado para mal las diferentes vulnerabilidades en esos sistemas que con mucha anticipación expertos habían señalado y de esta forma reforzar la afirmación hecha en la introducción de este trabajo: Las autoridades y los fabricantes implicados en elecciones con urnas electrónicas tienden a subestimar la complejidad real de realizar votaciones por ese medio.

Incidentes con urnas Electrónicas.

- 7/Abril/2003
  Virus interrumpe conteo en votaciones electrónicas del estado de Illinois
  Referencia: SANS NewsBites Abril 9, 2003 Vol. 5, Num. 14

- 25/Julio/2003
  Investigadores de Johns Hopkins University y Rice University descubren vulnerabilidad en software de la empresa Diebold que de ser explotado permitiría votar múltiples veces, cambiar los votos de otros votantes así como dar de baja el sistema de votaciones de manera prematura. La empresa argumentó que los investigadores evaluaron una vieja versión del software que nunca se usó en votaciones reales.

  Referencia: Rubin [47]

- Octubre 2003
  En el condado de Alameda California, se registró un inusitado record de participación; aunque posteriormente se comprobó que el software funcionaba correctamente, no se pudo verificar si el resultado era correcto.
  Referencia: Torres Wilbert [51]

- 4/Noviembre/2003
  En Boone County Indiana, los resultados iniciales de la elección reportaron más de 144,000 votos emitidos en una máquina especial de votación vía electrónica cuando una lista de votantes no rebasaba las 19,000 personas registradas.

  Misma fecha, Fairfax, Virginia, un conjunto de 10 máquinas dejaron de funcionar justo el día de la elección. Los republicanos impugnaron el resultado después de saber que las máquinas fueron retiradas del centro de votaciones para repararlas, acción que contrariaba la ley.
  Referencia: Torres Wilbert[51]

- 11/ Noviembre/ 2003
  En California se suspende la certificación de máquinas de votaciones de la empresa Diebold, por presunta instalación de software no certificado.
  Referencia: SANS NewsBites Noviembre 12, 2003 Volume: 5, Issue: 45

- Noviembre 2004
  Un incidente con una urna en Ohio causó que un candidato recibiera 3,893 más votos de los que debió tener; además hubo reportes de electores que tuvieron problemas para poder votar por cierto candidato.

  En Carolina del Norte más de 4,500 votos se perdieron por problemas con las limitaciones de la tarjeta de memoria.
  Referencia: SANS NewsBites Nov. 10, 2004 Vol. 6, Num. 45

- 17/Diciembre/2004
  La Empresa Diebold, tendrá que pagar al Estado de California 2.6 millones de dólares y 100,000 al condado de California, al ser demandado por actitudes fraudulentas sobre la seguridad de sus productos.
  Referencia: SANS NewsBites Dec. 22, 2004 Vol. 6, Num. 51

Consideraciones sobre Urnas Electrónicas:

A pesar del cuidado que se ha querido tener en el empleo de urnas electrónicas en Estados Unidos de Norteamérica, creando organismos encargados de legislar, certificar y establecer estándares sobre dichos sistemas; esto no ha sido suficiente, las irregularidades se siguen presentando y existen debilidades en este esquema que siguen sin tratarse adecuadamente.

Entre los aspectos que menos son cuidados se encuentra el tomar medidas para que los sistemas sean capaces de soportar los ataques de las autoridades y de sus propios

desarrolladores que en vez de tratar salvaguardar la legalidad del proceso, busquen el corromperlo.

Afortunadamente, las experiencias desfavorables también han ayudado a que las autoridades electorales mejoren sus procedimientos a fin de realizar votaciones con urnas electrónicas de forma más segura. Por ejemplo, actualmente existen fabricantes que envían el software de sus productos a las autoridades electorales, quienes al tener en su poder la versión oficial del software que la urna utilizará, cuentan con un medio de asegurarse que éste no sea alterado durante las elecciones.

Entre otras aportaciones que surgieron a raíz de los problemas suscitados, está el que varios fabricantes de urnas electrónicas ya contemplan la impresión de un comprobante de voto que permita auditar el proceso.

Por último ante las anomalías que han ocurrido con el complejo software que tienen que utilizar las urnas electrónicas para la interfaz gráfica y el audio que incorporan, han surgido urnas que no son más que transistores con código ensamblador incorporado en sus circuitos; estos sistemas buscan que su simplicidad de funcionamiento sea su fortaleza contra ataques externos e internos, dejando que sean los procedimientos y las personas las que den seguridad a su utilización (ver [13]).

1.2.6 Sistemas de Votaciones en Internet

El auge de Internet en el mundo, y del comercio electrónico, han generado el deseo de realizar también votaciones por medio de Internet. Así, de la misma manera que la tecnología ha permitido realizar compras desde el trabajo o el hogar; resultaría muy conveniente para los electores ser capaces de ejercer su voto sin tener que asistir a algún sitio en particular.

La mayoría de quienes proponen este tipo de sistemas de votaciones argumentan que se incrementaría la participación en estos procesos, sobre todo entre jóvenes, militares, marinos, viajeros, etc. Además de que en el largo plazo realizar estos comicios sería menos costoso. Sin embargo, este tipo de sistema de votaciones no ha sido probado aún.
(Report of the National Workshop on Internet Voting[41]).

Características

Es un Sistema de votaciones que hace uso de la tecnología de Internet para poder realizar elecciones. Todos los componentes que permiten que el votante se valide, ejercite su voto y que este último pueda ser almacenado para su conteo, son recursos remotos a los que se accede por medio de una red de computadoras.

Su principal característica y ventaja fundamental es que estos sistemas pueden utilizar la infraestructura ya existente en Internet, por lo que no necesitan encontrarse en un módulo de votaciones.

Tienen un gran nivel de accesibilidad haciendo factible que un votante pueda votar desde su domicilio. Además proveen como otros sistemas de votaciones electrónicos un excelente nivel de exactitud y velocidad en el conteo de votos.

Entre las tres modalidades para realizar Votaciones por Internet señaladas en Report of the National Workshop on Internet Voting [41] encontramos:

Votaciones en Módulos

Los votantes ejercen su voto desde lugares físicos bien definidos en los que las autoridades electorales administran la infraestructura de red y los equipos que se utilizan para votar. Lo anterior permite que se puedan aplicar ciertos controles sobre la plataforma de votaciones como la revisión del software que es instalado en las máquinas desde las cuales los electores votarán y el filtrado del tráfico externo a la red de comunicaciones que se utiliza para el sistema de votaciones.

Ventajas

Este tipo de votaciones por Internet es el más factible de llevarse a cabo; el hecho de que las autoridades tengan controles sobre el ambiente en el que las elecciones se realizarán, permite poder manejar los riesgos de seguridad de este tipo de votaciones de forma favorable.

Desventajas

El sistema de votaciones pierde accesibilidad, característica que se consideraba una de las principales ventajas que aportaba este tipo de tecnología.

Se requiere que la comunidad votante cuente con una cultura informática a fin de que ésta pueda hacer uso del sistema de votaciones.

Votaciones en Kioscos

Esta es una modalidad semejante a la votación en módulos, la diferencia consiste en que los votantes en vez de tener que dirigirse a un módulo de votaciones, emiten su voto desde algún sitio público como una biblioteca, centro comercial o escuela. La idea es que sobre dichos sitios se apliquen controles que permitan mitigar los riesgos de seguridad que se puedan suscitar. Es un estado intermedio entre las votaciones en módulos y las votaciones remotas.

Ventajas

Mayor Accesibilidad qùe con las votaciones en módulos.

Con los controles adecuados se podrían mitigar los riesgos, haciendo de esta modalidad un esquema factible de llevarse a cabo.

Desventajas

Implicaría un reto mayor tener que controlar más ambientes de votaciones por lo que el despliegue de recursos humanos y financieros tendría que ser superior.

La comunidad votante debe contar con una mínima cultura informática.

Votaciones Remotas

Esta modalidad permitiría a cualquier elector ejercer su voto desde cualquier localidad desde la que pueda acceder a Internet

Ventajas

Gran Accesibilidad

Desventajas

Las tecnologías actuales y las que se proyectan al corto y mediano plazo, no parecen resolver los grandes riesgos que la plataforma de Internet tiene. Por lo que muchos estudiosos del tema consideran a esta modalidad inviable. (ver Rivest[43]).

La comunidad votante debe contar con una adecuada cultura informática.

Procedimiento para votar por Internet

En un Sistema de Votaciones por Internet el siguiente procedimiento es desarrollado para realizar una elección.

1. El votante recibe de una manera segura la información necesaria para tener acceso al sitio electrónico donde podrá votar.

2. El votante entra al sitio electrónico y proporciona la información necesaria para su autenticación.

3. Le es presentado al votante el análogo de una boleta, donde el marca su decisión, y posteriormente confirma y envía su voto.

4. El sitio electrónico recibe y almacena el voto.

5. El conteo se realiza y los resultados son publicados.

Experiencia con las Votaciones en Internet

A pesar del gran auge que la integración de tecnologías en el Internet han tenido, las pruebas de los sistemas de votaciones bajo dicha plataforma son muy pocas.

Ésta tecnología no se ha utilizado en ambientes reales, casi toda la experimentación se ha llevado en ambientes académicos.

Las bases de mucho del trabajo académico en el área, se basa en un artículo de Fujioka, Okamoto y Ohta titulado "A practical secret voting scheme for large-scale elections" (ver [17]). Este artículo, promueve el uso de un protocolo que a través de firmas digitales ciegas, permite desligar al votante del voto que emitió.

Ejemplos de sistemas de votaciones que hacen uso del protocolo de votaciones de Fujioka son Sensus y E-Vox; para mayor información revisar Lorrie [10] y CIS [8] Respectivamente.

Entre algunas prácticas fuera del ambiente académico se encuentran las siguientes:

- Agosto de 1996 un partido político de Estados Unidos de Norteamérica utilizó un sistema por Internet (entre otros además de teléfono y correo postal), para elegir a su candidato, cerca de 2000 personas votaron por dicho medio.

- En otra elección privada, llevada en Arizona por el partido democrático, hubo un exitoso ejercicio con un sistema de votaciones por Internet en el que participaron 39,000 votantes.

Ambos ejercicios se llevaron a conclusión sin anormalidad alguna.

El aparente éxito en éste tipo de prácticas, ha incitado a continuar con experimentaciones con estos sistemas a fin de realizar una prueba en elecciones públicas.

Entre el intento real más reciente de llevar a elecciones publicas un sistema de votaciones en Internet ocurrió en Estados Unidos, el sistema se llamaba SERVE. Desafortunadamente y a pesar de los avances que en materia de votaciones en el ambiente académico existían, este sistema liberado a inicios del 2004, protegía de forma burda la confidencialidad de los votantes, y fue rechazado tras un informe publicado por especialistas en el tema ver Jefferson [27].

A pesar de que no se han probado a fondo los sistemas de votaciones por Internet, si se tienen los controles de ambiente adecuado, existen razones para pensar que puede llegar a ser uno de los sistemas de votaciones más confiables al reutilizar tecnología cuyos alcances son conocidos, además de ser sistemas auditables que pueden proporcionar verificabilidad parcial.

Por otro lado, la poca practicidad con que cuenta un sistema de votaciones en Internet al requerir a los votantes una cultura informática para su utilización, es un obstáculo insalvable, que impide pensar que un sistema de ésta naturaleza pueda ser implementado en gran escala sobre todo bajo condiciones sociales y económicas como las que imperan en nuestro país. Esto restringe la utilización del sistema propuesto a ser utilizado sólo en elecciones de menor escala donde los electores involucrados cuenten con el perfil adecuado.

En este trabajo se propone un sistema de votaciones en Internet, orientado a Votaciones en Módulos. A lo largo de este trabajo se detallarán los fundamentos sobre los cual su seguridad se soporta así como los aspectos de diseño e implementación. La factibilidad de su empleo bajo las diferentes modalidades posibles y la sugerida se abordará a detalle en el apartado de Conclusiones.

Capítulo 2 Diseño del Sistema de Votaciones

En este capítulo se expone el diseño del Sistema de Votaciones propuesto en este trabajo, para ello, se describe su protocolo, se abordan las tecnologías utilizadas para su implementación y se describe su modelo arquitectónico.

Cabe señalar que debido a que dentro de los objetivos de esta tesis no se encuentra el ser una referencia técnica de criptografía, se asumirá que el lector cuenta con conocimientos sólidos sobre dicha disciplina; especialmente en los principios que atañen a la operación de los sistemas criptográficos basados en curvas elípticas.

2.1 Protocolo de Votaciones

Existen diversos protocolos de votaciones que permiten cumplir con los requerimientos necesarios de los sistemas de votaciones (ver apartado 1.1). Sin embargo muchos parten de condiciones ideales, y son estas las que le otorgan cierto nivel de seguridad a su correcto funcionamiento.

La seguridad que puede proporcionar un protocolo de votaciones reside en su capacidad para poder enfrentar los posibles ataques que se realicen sobre él para alterar los resultados. De esta forma, un protocolo ideal debería ser capaz de soportar ataques por parte de los votantes y de las personas ajenas a un sistema de votaciones; así como también aquellos ataques internos por parte de las autoridades encargadas de las votaciones. Sin embargo, dicho protocolo aún no existe y no se espera que en un futuro cercano existan avances significativos que cambien ese hecho.

La razón de la anterior afirmación, es que a pesar de que es posible tener un buen grado de certidumbre de que por medio de técnicas criptográficas, la confidencialidad e integridad del voto así como el anonimato del votante, se pueda mantener frente a ataques por parte de votantes y de otras personas ajenas a la elección. Para el caso de ataques internos por parte de las autoridades, no hay un protocolo que brinde una defensa sólida. Ante este obstáculo se han adaptado los protocolos a fin de dividir los roles de tal forma que la responsabilidad de las votaciones no se sostenga en una sola entidad, logrando así que un ataque sea exitoso sólo si más de una entidad del protocolo colude para ello. Esta estrategia si bien ha brindado confianza a los protocolos de votaciones, no es de ninguna manera una solución contundente ante la posibilidad de un ataque por parte de las autoridades electorales.

En general, los protocolos de votaciones buscan de alguna manera separar la identidad del Votante del voto que éste emitió. Para ello el protocolo del sistema de Votaciones propuesto hace uso de firmas digitales ciegas, las cuales se describirán a continuación.

Firmas Digitales Ciegas.

En este esquema de firmas digitales se identifican dos entidades, quien solicita la firma (cliente) y quien realiza la firma digital (firmante). La idea detrás de este tipo de firmas es que el cliente pueda obtener una firma digital válida sobre un mensaje, pero sin que el firmante, una vez terminado el proceso, conozca el mensaje que firmó y la firma digital asociada a éste.

Para que lo anterior se pueda llevar a cabo, el mensaje es transformado (enmascarado) y se envía al firmante para que genere la firma de ese mensaje alterado; el resultado de firmar dicho mensaje es la obtención de una firma digital ciega. Una vez obtenida la firma, el cliente le aplica una transformación (desenmascarado) a fin de obtener una firma digital normal que permita verificar la autenticidad del mensaje original.

Estos esquemas cobran importancia en transacciones financieras en los que hace falta obtener autorización de una entidad protegiendo la confidencialidad de lo que se solicita. Otra aplicación práctica de las firmas digitales ciegas, que tiene trascendencia en este trabajo, es su utilidad en los protocolos de votaciones.

2.1.1 Protocolo de FOO (Fujioka, Okamoto y Ohta)

Debido a que el protocolo del sistema de votaciones propuesto se basa en uno diseñado por Fujioka, Okamoto y Ohta publicado en el artículo "A practical secret voting scheme for large-scale elections" (Ver [17]); se comenzará con la explicación de éste para posteriormente abordar a detalle el protocolo utilizado en la implementación del Sistema de Votaciones Propuesto.

El Protocolo de FOO, contempla dos entidades un Autenticador y un Contador. La entidad Autenticador, se encarga de verificar la identidad de los votantes y el Contador, es la entidad que almacenará los votos y realizará el conteo de estos.

Abreviaturas utilizadas

E( )	Función de cifrado con criptosistema simétrico
D( )	Función de descifrado con criptosistema simétrico
B( )	Función de enmascarado de un mensaje
$B( )^{-1}$	Función de desenmascarado de un mensaje
v.	voto
vc	voto cifrado
bvc	voto cifrado preparado para obtener firma digital ciega
fdv	Firma Digital del Votante
fda	Firma Digital del Autenticador
fdab	Firma Digital del Autenticador Enmascarada
kv	clave para cifrar el voto.
Fg( )	Algoritmo de generación de Firma Digital
Fv( )	Algoritmo de verificación de Firma Digital

C	Contador
A	Autenticador
Vo	Votante
Id	Identificador del Votante
Vprk	Llave privada del Votante
Vpuk	Llave pública del Votante
Aprk	Llave privada del Autenticador
Apuk	Llave pública del Autenticador

Supuestos

- El Autenticador y cada uno de los votantes que participará generan una llave pública y una privada.
- El Autenticador recibe de forma segura la llave pública de cada uno de los Votantes
- Los Votantes reciben de forma segura la llave pública del Autenticador.

Pasos del Protocolo

1. El Votante cifra su voto con un criptosistema simétrico y una clave propia, generando un voto cifrado

   $vc = E(v, kv)$

   El votante genera una máscara al voto cifrado para obtener una firma digital.

   $bvc = B(vc)$

   El Votante genera con su llave privada una firma digital al voto cifrado enmascarado.

   $fdv = Fg(bvc, Vprk)$

   Finalmente el Votante envía el voto cifrado y enmascarado junto con su identificador y la firma digital al Autenticador

   $Vo \rightarrow A: fdv, bvc, Id$

2. El Autenticador verifica la firma digital

   $?Fv(bvc, fdv, Vpuk)$

Si la firma es valida, el Autenticador sabe que el Votante es válido, si éste no ha votado, firma el voto cifrado y enmascarado

fdab= Fg(bvc,Aprk)

Por último envía la firma digital al Votante

A→Vo:fdab

3. El Votante recibe la firma digital que le fue enviada por el Autenticador y verifica que sea una firma valida para el voto cifrado y enmascarado

   ? Fv(bvc, fdab,Apuk)

   Si la firma es válida, se desenmascara la firma

   $fda=B(fdab)^{-1}$

   Finalmente se envía la firma digital del voto cifrado al Contador junto con la firma digital obtenida del autenticador.

   Vo→C: vc,fda

4. El Contador verifica que la firma digital del voto cifrado sea valida

   ?Fv(vc,Fda,Apuk)

   Si la firma digital es valida, ésta y el voto cifrado es almacenado.

5. Terminada la elección el Contador publica la lista de los votos cifrados junto con las firmas digitales que fueron validas. El Votante observa que su voto se encuentre en la lista y proporciona la clave secreta con la que éste fue cifrado

   Vo→C: kv

   El Contador descifra los votos y los almacena.

6. Una vez que se han descifrado todos los votos, el Contador cuenta los votos y publica los resultados.

7. Por último el Contador, publica junto con los resultados, los votos en texto claro, los votos cifrados, las llaves con las que descrifró los votos y la firma digital asociada a estos. A su vez el Autenticador publica la lista de Votantes que se registraron y las firmas digitales ciegas que emitió.

Observaciones

a) Votos alterados, eliminados e inválidos pueden ser detectados.
b) Este protocolo proporciona verificabilidad parcial ya que sólo el votante conoce la firma digital ciega que obtuvo y su relación con la firma digital que envía al Contador.
c) Este protocolo garantiza un buen nivel de confidencialidad
d) La Seguridad del sistema recae en el autenticador quien de ser corrupto esta en posibilidad de generar votos por los que se abstuvieron de votar.
e) Involucra el complejo problema de Administrar un par de llaves, publica y privada, para cada uno de los votantes.

Consideraciones finales sobre el Protocolo de Votaciones FOO

Como se puede observar una de las características más importantes de los protocolos de votaciones, es que sean capaces de separar de forma efectiva la identidad del votante del voto que éste emitió. El protocolo de FOO, se tiene considerado como uno de los más robustos precisamente por su capacidad para hacerlo por medio de firmas digitales ciegas. Por esa razón, en él se han basado la mayoría de los proyectos que se han hecho a nivel académico en el campo de sistemas de voto por Internet. Sin embargo, cabe hacer hincapié que el problema de generar, distribuir y administrar un par de llaves publica y privada para cada uno de los votantes, y el hecho de que estos tengan además que generar una clave secreta, vuelven a este protocolo extremadamente difícil de utilizarse en elecciones reales.

El protocolo que se utiliza en el sistema propuesto, esta basado en FOO con ciertas modificaciones; se busca aprovechar la seguridad de él, pero restándole complejidad a fin de que su implementación sea posible. Los detalles del protocolo que el sistema propuesto utiliza se describen a continuación.

2.1.2 Descripción del protocolo del Sistema de Votaciones

A continuación se describirá detalladamente el protocolo utilizado para la implementación del sistema de votaciones; si bien conserva la esencia del protocolo FOO, se realizaron las siguientes modificaciones substanciales:

A. Para facilitar la administración de las credenciales de los votantes, en vez de proporcionarles un par de llaves pública y privada, su autenticación la realizan por medio de una contraseña y un id de usuario.
B. Se brinda mayor seguridad en el caso de que el Autenticador (denominado Verificador en este capítulo) pretenda generar votos ilegales.
C. Se realizan las modificaciones pertinentes para poder realizar las firmas digitales ciegas por medio del esquema ECDSA (Elliptic Curve Digital Signature Algorithm).

D. Se limita la verificación por parte del Votante, a fin de que él obtenga la información de que su voto fue recibido por el Contador, pero no se pueda demostrar a terceros por quien votó. Esto con el objetivo de evitar la coerción.

Observaciones Previas

Se notará que se cambió el nombre de Verificador por el de Autenticador; el Verificador desempeña las mismas funciones que la entidad denominada Autenticador del capítulo anterior. La razón de renombrar esa entidad fue el estandarizar los nombres utilizados en el protocolo con el de otros elementos que fueron definidos para la implementación del sistema y que serán detallados en el siguiente capítulo.

A su vez cabe mencionar que los detalles de las funciones de generación y verificación de firmas digitales así como las de enmascaramiento y desenmasacarado de un mensaje para obtener una firma digital ciega, serán tratados en una sección posterior en este mismo capítulo.

Entidades del Sistema de Votaciones

Los siguientes roles son definidos en el protocolo de votaciones a fin de proporcionar las características de Confidencialidad y Confiabilidad que se definieron previamente.

Verificador:

Programa responsable de la autenticación de los usuarios, es capaz de leer la lista de votantes válidos en donde coteja identificador de votante y contraseña. Además tiene privilegios de escribir en dicha lista cuando el votante ha ejercido su voto; a su vez es el responsable de generar la firma ciega que le dará validez al voto. Cuenta con un par de llaves pública y privada. Es el análogo a la entidad autenticador en el protocolo descrito previamente.

Contador:

Programa responsable de verificar la validez de los votos y de almacenarlos; a su vez será el responsable de contar los votos una vez que el periodo de votaciones haya finalizado, cuenta con la llave pública del Verificador.

Votante:

Usuario del Sistema de Votaciones con derecho a ejercer su voto.

Pasos del Protocolo del Sistema de Votaciones

Abreviaciones utilizadas en la Descripción del Protocolo

Vr	Verificador
VrPrk	Llave Privada del Verificador
VrPuk	Llave Pública del Verificador
Cr	Contador
Ve	Votante
id	Identificador de elector
p	contraseña
H()	Función Hash
B()	Función de enmascarado de un mensaje
$B()^{-1}$	Función de desenmascarado de un mensaje
Fg()	Algoritmo de Generación de Firma Digital
Fv()	Algoritmo de Verificación de Firma Digital
v	voto en texto plano
hv	hash del voto
hvb	hash del voto enmascarado
m	primer número pseudoaleatorio
n	segundo número pseudoaleatorio
fhv	firma digital sobre el voto
fhvb	firma digital sobre el voto enmascarado
R	Punto Aleatorio sobre una curva elíptica

Supuestos

- Es requisito de este protocolo que toda la comunicación que se realice entre las tres entidades que integran el protocolo se realice a través de un canal seguro, esto a fin de evitar ataques activos o pasivos a la comunicación.

- A fin de hacer más comprensible el protocolo, se omitirán aquellos pasos relacionados con el establecimiento del canal seguro. Los detalles sobre el establecimiento de éste serán revisados en una sección posterior de este capítulo.

- Se parte del supuesto de que la entidad Verificador ya generó su par de llaves pública y privada.

- Se da por hecho que la entidad Contador ya cuenta con la llave pública del Verificador.

- Se da por hecho que a cada Votante que participe se le entrega de forma confidencial su identificador de Votante y su contraseña.

- En la lista que está en poder del Verificador donde coteja identificador de usuario y contraseña, no se encuentra la contraseña sino el hash de ésta; esto a fin de que el Verificador desconozca las contraseñas.

Pasos del Protocolo

1. El Votante envía sus credenciales al Verificador (identificador de elector y contraseña)

$Ve \rightarrow Cr : id,p$

2. Si las credenciales son válidas y el votante no ha votado aún, el Verificador envía su llave pública y un punto aleatorio R como insumo para que se realice la firma digital ciega con ECDSA. En caso de que el votante no haya proporcionado credenciales válidas o ya haya votado, se interrumpe la comunicación con él.

$Vr \rightarrow Ve: R, VrPuk$

3. El votante genera su voto. Su voto es concatenado con un número pseudoaleatorio y se usa como entrada de una función hash; el resultado es concatenado con otro número pseudoaleatorio y se vuelve a procesar en una función hash. Al resultado le llamaremos hash del voto.

$hv= H(n,H(v,m))$

Hecho esto, se añade una máscara al hash del voto a fin de poder obtener una firma ciega del Verificador.

$hvb=B(hv)$

Finalmente El votante envía el hash del voto enmascarado al Verificador

$Ve \rightarrow Vr: hvb$

4. El Verificador recibe el hash del voto enmascarado, y genera una firma digital con su llave privada obteniendo la firma ciega de dicho mensaje.

$fhvb=Fg(hvb,VrPrk)$

Finalmente el Verificador envía la firma digital del hash del voto enmascarado al Votante. Para fines de auditoria, almacena la firma digital que generó y el password que recibió.

$Vr \rightarrow Ve: fhvb$

5. El Votante recibe la firma digital ciega del hash del voto enmascarado y verifica que sea correcta.

? Fv(hvb, fhvb, VrPuk)

De ser correcta el Votante realiza el desenmascarado de la firma digital del hash del voto.
$fhv = B(fhvb)^{-1}$

Hecho lo anterior el votante envía su voto en texto plano, la firma digital del hash de su voto, y los números aleatorios con los que se computa el hash del voto al Contador.

Ve → Cr: fhv,v,m,n

6. El Contador recibe los cuatro parámetros enviados por el votante, y computa el hash del voto.

hv= H(n,H(v,m))

Hecho lo anterior el Contador verifica que la firma digital del hash del voto sea correcto.

fhv=?Fv(hv,VrPuk)

Si el voto es válido, el Contador lo almacena

7. Terminado el periodo para realizar las votaciones, el Contador realiza el escrutinio de los votos.

8. Una vez terminado el conteo de los votos. El Contador publica el resultado de las elecciones y una lista donde incluye todas las firmas digitales de los votos que recibió las *fhv* de todos los participantes.

A su vez el Verificador hace pública las firmas que generó y a que votantes las asignó, junto con los passwords en claro que recibió. *id,fhvb,p*

Para una explicación gráfica de estos pasos ver el Diagrama 1.

DIAGRAMA DEL PROTOCOLO DEL SISTEMA DE VOTACIONES

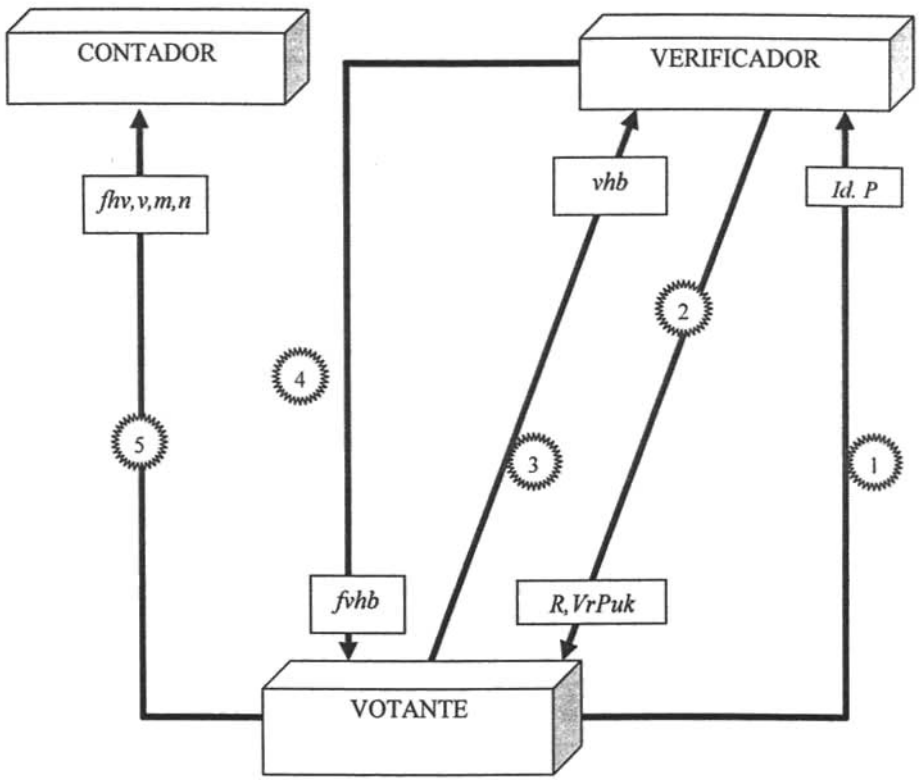


Diagrama 1

2.1.3 Esquema de Firmas Digitales usada por el Sistema de Votaciones

ECDSA

Entre los esquemas para realizar firmas digitales con criptografía de curvas elípticas, se encuentran el ECDSA (Elliptic Curve Digital Signature Algorithm) y el EC-KCDSA (Elliptic Curve Korean Certificate Digital Signature Algorithm); siendo ECDSA el más estandarizado al aparecer en el ANSI X9.62, FIPS 186-2, IEEE1363-2000 e ISO/IEC 15946-2, razón por la cual se utilizó para la implementación del sistema que se propone.

ECDSA es una firma digital con apéndice, por lo cual para realizar la verificación de una firma digital de este tipo, resulta necesario contar con el mensaje original.

El Algoritmo para poder realizar una firma digital bajo dicho esquema es el siguiente:

Algoritmo de ECDSA

Generación de Llaves

Cada entidad **A** debe realizar los siguientes pasos a fin de generar las llaves que utilizará el criptosistema:

1. Seleccionar una curva elíptica **E** definida sobre **Zp,** el número de puntos en $E(Z_p)$ deben ser divisibles por un primo grande **n**.
2. Seleccionar un punto $P \in E(Z_p)$ de orden **n**
3. Seleccionar un entero **s** único e impredecible en el intervalo **[1,n-1]**
4. Calcular **Q=sP**
5. La Llave pública de **A** es **(E,P,n,Q);** y su llave privada sería **s**

Generación de Firma Digital
Para firmar un mensaje m, A realiza los siguientes pasos:

1. Seleccionar un entero aleatorio **k** en el intervalo **[1, n-1]**.
2. Calcular $kP= (x_R,y_R)$, $c = x_R \bmod n$ (Aquí x_R es tratado como un entero)

   Si **c=0** regresar al paso 1. (Esta es una condición de seguridad: si **c = 0**, entonces la ecuación $d = k^{-1} \{e+ sc\} \bmod n$ no involucra a la llave privada **s**
3. Calcular $k^{-1} \bmod n$
4. Calcular **e = h(m),** en el que **h()** es un Algoritmo de Función Hash considerado seguro y $d = k^{-1} \{e + sc \} \bmod n$
5. Si **d = 0** entonces ir al paso 1 (si **d = 0**, entonces $d^{-1} \bmod n$ no existe, d^{-1} es requerido en el paso 2 de la verificación de la firma)
6. La firma del mensaje m es el par de enteros **(c,d)**

Verificación de la Firma digital

Para Verificar la firma digital de **A (c,d)** asociado al mensaje **m**, **B** realiza los siguientes pasos:

1. Obtener una copia auténtica de la llave pública de **A (E,P,n,Q)**
2. Verificar que **c** y **d** son enteros en el intervalo **[1,n-1]**
3. Calcular $\mathbf{w = d^{-1} \bmod n}$ y **e=h(m)**
4. Calcular $\mathbf{u_1 = ew \bmod n}$
5. Calcular $\mathbf{u_2 = cw \bmod n}$
6. Calcular $\mathbf{R'(x_{R'},y_{R'}) = u_1P+u_2Q;\ v=x_{R'} \bmod n}$
7. Aceptar la firma solo si **v = c**

Comprobación

Los términos **w**, $\mathbf{u_1}$, $\mathbf{u_2}$ pueden ser reescritos de la siguiente manera:

$$w = k(e + sc)^{-1}$$

Cabe mencionar que aquí $\mathbf{e_1}$, representa al hash del mensaje generado por quien quiere verificar la firma digital; y es diferente de **e**, éste último creado durante la generación de la firma digital

$$u_1 = e_1 k(e + sc)^{-1}$$

$$u_2 = ck(e + sc)^{-1}$$

Reescribiendo en términos de la ecuación de verificación (Verificación de Firma Digital paso 6)

$$R' = e_1k(e + sc)^{-1} P + sck(e+sc)^{-1} P$$

Reagrupando

$$R' = (e + sc)^{-1} (e_1kP + sckP)$$

Factorizando por $(e_1 + sc)$

$$R' = (e + sc)^{-1} (e_1 + sc) kP$$

Sólo si son idénticos **e** y $\mathbf{e_1}$, el único término que quedará será **kP**, el cual nos genera el punto **R** original. Bajo esas condiciones la firma demostraría ser valida tras realizar la comparación pertinente de **v** y **c**.

Algoritmo de ECDSA para calcular una Firma Digital Ciega

Para poder utilizar ECDSA a fin de obtener una firma digital ciega, es necesario realizar modificaciones sobre los pasos de generación y verificación de la firma. Cabe señalar que los pasos para generar las llaves de las que hace uso ECDSA no se ven modificados. A continuación se describirán los pasos con las modificaciones pertinentes.

Pasos previos a la Generación de la Firma

Antes de que el cliente solicite la firma, el firmante debe generar un punto aleatorio en la curva elíptica que le será enviado al cliente.

Del lado del Firmante

a) Generar entero **k** en el intervalo **[1, n-1]**.
b) Calcular el punto **Rc = kP**. Si $\mathbf{x_{Rc} \equiv 0 \mod n}$. regresar al paso anterior
c) el Firmante envía al cliente que solicita la Firma Ciega el punto **Rc**

Del lado del Cliente

El cliente recibe el punto que le fue enviado, lo utiliza como base para calcular otro que le permitirá enmascarar el resultado de la función hash aplicado a su mensaje.

d) Genera entero α en el intervalo **[1, n-1]**.
e) Calcula el punto $\mathbf{R = \alpha Rc}$
f) Calcula la máscara $\beta = x_{Rc}\, \alpha\, x_R^{-1}$
g) Calcula **e=h(m)**
h) Calcula hash enmascarado $\mathbf{e' = \alpha^{-1} \beta\, e}$
i) Se envía al firmante **e'**

Generación de Firma

En este apartado identificaremos al par **(d', c')** como la firma digital sobre el hash enmascarado e'; diferenciándolos del par **(d , c)** que vendría a ser la firma digital sobre el hash e que corresponde al mensaje original.

1. Tomar el componente **x** del punto **Rc** como primer componente de la firma.
   $\mathbf{c' = x_{Rc} \bmod n}$

2. Calcular $\mathbf{d' = k^{-1}(e' + sc')}$
   Si $\mathbf{d' \equiv 0 \bmod n}$. Se tendrá que reiniciar todo el proceso desde los pasos previos a la generación de la firma.

3. La Firma Digital Ciega de **m** es **(c', d')**, la cual es verificable para **e'**

Desenmascarado de la Firma Digital Ciega

Del lado del Cliente

Se recibe la firma digital sobre el elemento enmascarado.

1. Verificar que **(c',d')** es válido para **e'**

Este paso se realiza de la misma forma en la que se verifica una firma digital normal creada bajo el esquema de ECDSA

2. Desenmascarar la Firma Digital
   $\mathbf{d = d' \beta^{-1}}$
   $\mathbf{c = X_R}$
3. La firma Digital para **m** es **(c,d)**

Verificación de Firma (de la misma manera que se explicó previamente)

1. Obtener una copia auténtica de la llave pública de **A (E,P,n,Q)**
2. Verificar que **c** y **d** son enteros en el intervalo **[1,n-1]**
3. Calcular $\mathbf{w = d^{-1} \bmod n}$ y **e=h(m)**
4. Calcular $\mathbf{u_1 = ew \bmod n}$
5. Calcular $\mathbf{u_2 = cw \bmod n}$
6. Calcular $\mathbf{R'(x_{R'},y_{R'}) = u_1P+u_2Q;\ v=x_{R'} \bmod n}$
7. Aceptar la firma solo si **v = c**

Comparar el componente ***x*** de ***R'*** contra el valor **c** de la firma digital

La firma es válida sólo si la congruencia $\mathbf{x_{R'} \equiv x_R \bmod n}$ es verdadera.

Comprobación

Si expandimos las expresiones cuando se quita el desenmascarado, podemos observar como se eliminan los términos que se añaden, hasta quedar una firma digital del esquema ECDSA normal para el mensaje m.

$d' = k^{-1}(e' + sc')$

Expandiendo e' y c' en la firma digital enmascarada

$= k^{-1} (\alpha^{-1} x_{Rc} \alpha x_R^{-1} e + s x_{Rc})$

Multiplicando por β^{-1}, el equivalente a quitar la mascara a la firma digital

$= k^{-1} (\alpha^{-1} x_{Rc} \alpha x_R^{-1} e + s x_{Rc}) (x_{Rc}^{-1} \alpha^{-1} x_R)$

$= k^{-1} (\alpha^{-1} x_{Rc} \alpha x_R^{-1} e \; x_{Rc}^{-1} \alpha^{-1} x_R + s x_{Rc} x_{Rc}^{-1} \alpha^{-1} x_R)$

Reordenando notamos que muchos términos pueden ser eliminados

$= k^{-1} (\alpha^{-1} \alpha x_R^{-1} x_R \; x_{Rc}^{-1} x_{Rc} e \alpha^{-1} + s x_{Rc} x_{Rc}^{-1} \alpha^{-1} x_R)$

$= k^{-1} (e\alpha^{-1} + \alpha^{-1} s x_R)$

Entonces la firma válida para e, es

$d = \alpha^{-1} k^{-1} (e + s x_R)$

$c = x_R$

Esta firma digital ya tiene los mismos términos que una firma digital normal.

Para que el firmante pueda asociar la firma digital que generó de la que obtendrá el cliente después de quitar el enmascarado, tendría que calcular el valor **α**, este valor fue por el que se multiplicó a **Rc**, para generar **R**. Esto significa, que tendría que resolver el problema del logaritmo discreto elíptico; problema intratable si los bits con los que opera el criptosistema son los adecuados (160 bits mínimo).

2.1.4 Análisis de Seguridad del Protocolo

Seguridad del Sistema

Exactitud del sistema

Debido a las propiedades de las firmas digitales, se evita que los votos sean alterados o sean almacenados para el conteo de votos que no fueron emitidos por los votantes.

En caso de que la llave privada del Verificador fuera comprometida por un atacante, el incidente sería detectado en una revisión posterior al encontrarse más votos recibidos por el Contador que aquellos que fueron firmados por el Verificador.

En caso de que el Verificador utilizara su llave privada para generar firmas ciegas que nunca fueron solicitadas y así enviar votos al Contador, también tendría que conocer las contraseñas de los usuarios para evitar ser detectado, información de la que carecería pues en la lista de Votantes que utiliza solo se encuentran los hash de las contraseñas.

Privacidad

Los números aleatorios con los que es computado el hash del voto permiten que se pueda obtener la firma ciega sin enviar el voto.

El que el hash del voto esté compuesto por dos números pseudoaleatorios y la doble aplicación de una función hash; hace computacionalmente difícil que un verificador corrupto trate de adivinar los votos que se están firmando. A su vez también previene una modificación posterior al voto por parte de un Contador corrupto.

La privacidad del votante se mantiene enviando por separado las credenciales que identifican al votante, del voto que emite. Esto es debido a que el hash del voto, el hash enmascarado del voto y las firmas digitales asociados a cada uno, no pueden ser vinculadas de forma trivial por las propiedades de las firmas ciegas previamente expuestas.

Verificabilidad

Por medio de este protocolo es posible obtener verificabilidad parcial sin comprometer la privacidad del Votante, esto se debe a que él es el único que conoce la firma digital ciega que fue enviada por el Verificador y la firma digital que se obtiene al desenmascarar ésta. Por lo anterior, el Votante puede verificar que el Verificador si haya registrado que votó, y que el contador haya recibido su voto.

Vulnerabilidades

A nivel de protocolo este sistema tiene el inconveniente de que si el Verificador y el Contador se coluden para vulnerar al sistema, podrían enviarse información sobre como van registrándose y como van llegando los votos al Contador. Si los votantes no utilizan el sistema en gran número, podrían fácilmente identificar por quien ha votado cada elector.

2.2 Estructura del Sistema de Votaciones

Al ser un Sistema de Votaciones por Internet el propuesto en este trabajo, no sólo se deben cubrir los requerimientos inherentes a los sistemas de votaciones, además se busca aprovechar la infraestructura de redes basadas en TCP/IP para la realización de elecciones de forma remota.

De entrada se puede mencionar que el sistema propuesto, se ejecutará de forma independiente en al menos tres computadoras que se encuentran en red. Una de ellas será para alojar a la entidad Contador, una más para el Verificador, y la tercera será para uso del Votante.

Ello implica que se debe determinar el protocolo de comunicaciones en el cual se realice el intercambio de información entre los Votantes y las demás entidades del sistema. A su vez se debe definir el lenguaje de programación bajo el cual la interfaz y las funciones criptográficas se implementarán así como el mecanismo de almacenamiento de los votos. Dichos tópicos serán abordados detenidamente a continuación

2.2.1 Plataforma de Comunicación del Sistema

Primeramente se determinó que en vez de implementar un esquema de comunicaciones propio, el sistema estaría orientado al WWW; por ésta razón todo el aspecto de comunicaciones se delegaría a un servidor Web.

Siendo que las entidades del sistema de votaciones se ejecutarían en equipos independientes, en ellos se encontraría ejecutándose a su vez un servidor web quien les proporcionaría comunicación con la red.

Ésta estrategia, le da mayor simplicidad al código y vuelve en general más sencillo al sistema; aspecto importante al momento de realizar una auditoria sobre el código.

Otras motivaciones consideradas no por ello menos importantes son las siguientes:

1) Las aplicaciones Web en general tienen gran accesibilidad debido a que cualquier usuario tiene acceso a ésta sin necesidad de instalar programas clientes en cada una de las máquinas que el usuario utilizará.

2) El Web tienen cerca de diez años de experiencia en el campo del comercio electrónico; esto ha permitido que la seguridad en aplicaciones de este tipo se desarrolle por lo cual actualmente se encuentran disponibles diversas tecnologías para robustecer la seguridad de los sistemas.

3) Los servidores Web tienen entre sus prioridades el ser seguros y el mitigar en lo más posible la explotación remota de vulnerabilidades al intercambiar información. Por esta razón, el mantenimiento de este tipo de programas es continuo.

4) La tecnología Web se integra fácilmente con SSL (Secure Socket Layer), protocolo de comunicación segura, basado en criptografía de llave pública que permite la obtención de un canal de comunicación seguro.

Cabe señalar que la utilización de servidores Web, también tiene una contraparte adversa debido a que se introduce el problema de tener que cuidar de su administración.
Por la complejidad del tema se recomienda ver Brittain[3] para involucrarse a fondo en la Administración del servidor Web Tomcat, tecnología elegida para implementar el Sistema de Votaciones.

El Servidor Web Tomcat

Tomcat surgido de los laboratorios de Sun Microsystems, es un servidor web estable y sólido; a su vez también es un servidor de servlets y JSPs (programas hechos en Java orientados a dar la funcionalidad de CGIs). Actualmente forma parte del proyecto Apache, lo que permite que su distribución y uso sea gratuita, además de que su código está abierto a revisión.

A pesar de que claramente el servidor Web más utilizado, con mayor soporte en funcionalidades y prestigio es Apache. El servidor Web en el que se implementó el sistema de Votaciones es Tomcat. Las razones en las que se justifica la decisión de utilizarlo son las siguientes:

1) Seguridad: Debido a que Tomcat corre sobre una máquina virtual situada entre la red y el sistema operativo, se previenen casi todos los tipos de ataques de buffer overflow. Por ello Tomcat no es tan susceptible a ataques remotos de ese tipo a diferencia de otros servidores Web escritos en lenguajes compilados como C o C++.

2) Migración: Dado que Tomcat está escrito en Java, la migración del servidor Web a otra arquitectura es muy simple; únicamente hay que copiar todo el árbol de directorios de tomcat y moverlos al nuevo servidor para que este sea completamente funcional.

3) Configuración: Tomcat es más fácil de configurar

4) Actualización: Se pueden instalar nuevas versiones de Tomcat sin repercusión alguna para la funcionalidad del contenido que se tenía hospedado.

A pesar de las ventajas de Tomcat, Apache es un servidor web más veloz, y con muchas más funcionalidades. Éste cuenta con una lista muy larga de módulos que se pueden integrar a él. Además existen paquetes de software de terceros que enriquecen su funcionalidad. Sin embargo, para la funcionalidad que requiere específicamente el Sistema de Votaciones que se propone, y por el lenguaje de programación que se eligió para desarrollar dicho el sistema, Tomcat es el Servidor Web más idóneo.

Canal Seguro

Para la obtención del canal seguro, se hace uso de SSL; éste es un protocolo de fácil integración al WWW que por medio de criptografía de llave pública y una PKI permite obtener servicios de autenticación, confidencialidad e integridad en un canal de comunicación.

El concepto de SSL fue creado en los laboratorios de Netscape; así como las primeras versiones; posteriormente surgió la necesidad de convertirlo en un estándar para lo cual el IETF estableció el TLS definido en el RFC 2246, éste vino a ser una actualización menor sobre el SSL v.3 de Netscape.

El protocolo SSL, es una capa que se sitúa entre la capa de transporte de TCP/IP y la capa de aplicación. Mientras el estándar TCP/IP simplemente envía información libre de errores sin autenticar entre dos computadoras, SSL además añade ciertas características de seguridad a ese flujo de datos.

- Autenticación del Servidor, utilizando firmas digitales
- Autenticación del cliente, por medio de firmas digitales
- Confidencialidad en los datos utilizando criptosistemas simétricos y asimétricos.
- Integridad en los datos por medio de funciones hash.

El establecimiento de un canal seguro por medio de SSL sigue los siguientes pasos:

i. El cliente accede al sitio Web del Servidor
ii. El servidor le presenta al cliente un certificado digital (del tipo X.509).
iii. El cliente presenta al servidor su certificado digital (opcional)

iv. Se gestiona una clave y un algoritmo de cifrado para emplearse en la comunicación de las dos entidades.
v. Comienza la comunicación segura cifrando con la llave y el criptosistema simétrico acordado.

La gran ventaja de utilizar SSL en las aplicaciones, es que le permiten ocultar al desarrollador de aplicaciones web, la complejidad de establecer un canal seguro. De esta forma el establecer un canal seguro generalmente se limitará a referenciar que la comunicación en vez de realizarse vía http se realice por https.

Una de las principales desventajas que tenía el uso de SSL eran sus limitaciones al tamaño de las llaves que se podían utilizar debido a restricciones en la exportación de aplicaciones criptográficas creadas en Estados Unidos; sin embargo, a partir de enero del 2000 estas restricciones se relajaron y es posible exportar criptografía libremente salvo a aquellos países en los que se considera que brindan apoyo a organizaciones terroristas. Para información actualizada acerca de restricciones aplicables a la criptografía, consultar http://www.bis.doc.gov/ , la página del "Bureau of Industry and Security U.S. Department of Commerce".

2.2.2 Base de Datos

Se consideró conveniente para la implementación de este proyecto, el utilizar una base de datos en vez de archivos de texto plano para almacenar los datos. Esto además de obedecer a fortalecer el medio de almacenamiento de los votos, proporciona alta accesibilidad a las listas de autenticación que utilizará el verificador.

La Base de Datos que se utilizó fue PostgreSQL, esta es la base de datos más robusta disponible en el rubro del software libre. Además de la conveniencia de no requerirse comprar una licencia para su uso, es equiparable en estabilidad a las bases de datos comerciales.

En evaluaciones las bases de datos comerciales consolidadas en el mercado como oracle, muestran una capacidad superior para soportar mayores volúmenes de información así como de usuarios concurrentes que realicen lectura y escritura sobre diversas bases de datos. Sin embargo, Por la naturaleza del proyecto, no se requiere de gran capacidad en la atención de transacciones de dicho tipo, y si de procesador debido a la tareas de generación y verificación de firmas digitales por lo que PostgreSQL satisface los requerimientos de el sistema propuesto.

PostgreSQL se ha utilizado como base de datos para aplicaciones reales, con buenos resultados; entre dichas empresas que se enumeran como casos de éxito se encuentra Verisign y Afilias, prestigiadas empresas que soportan la infraestructura de llave pública del Internet. Para mayor información sobre los casos de éxito con esta Base de Datos ver el informe de Dravis Group [11].

2.2.3 Lenguaje de Programación

Para poder realizar un bosquejo general del Sistema de Votaciones sólo hace falta identificar los módulos de programación que harán funcionar al sistema e integrarlos con los componentes anteriormente expuestos. Las capas en las que existe el requerimiento de realizar programación para el funcionamiento del Sistema de Votaciones del que trata este trabajo, se pueden identificar en tres diferentes:

1. Del lado de los servidores Web dónde se encontrarán las entidades Contador y Verificador, para la atención de peticiones, generación de firmas digitales, y autenticación del cliente.

2. Del lado del cliente donde se encontrará el votante, para la preparación del hash del voto con el que se obtendrá una firma digital ciega, verificación de la firma digital y envío del voto.

3. Entre los Servidores Web y la Base de Datos para el almacenamiento del Voto y las consultas a la lista de electores válidos.

Con base en los requerimientos de programación descritos, y los componentes descritos en secciones previas, es posible establecer un diagrama inicial de Sistema de Votaciones ver Diagrama 2

DIAGRAMA PREVIO DEL SISTEMA DE VOTACIONES

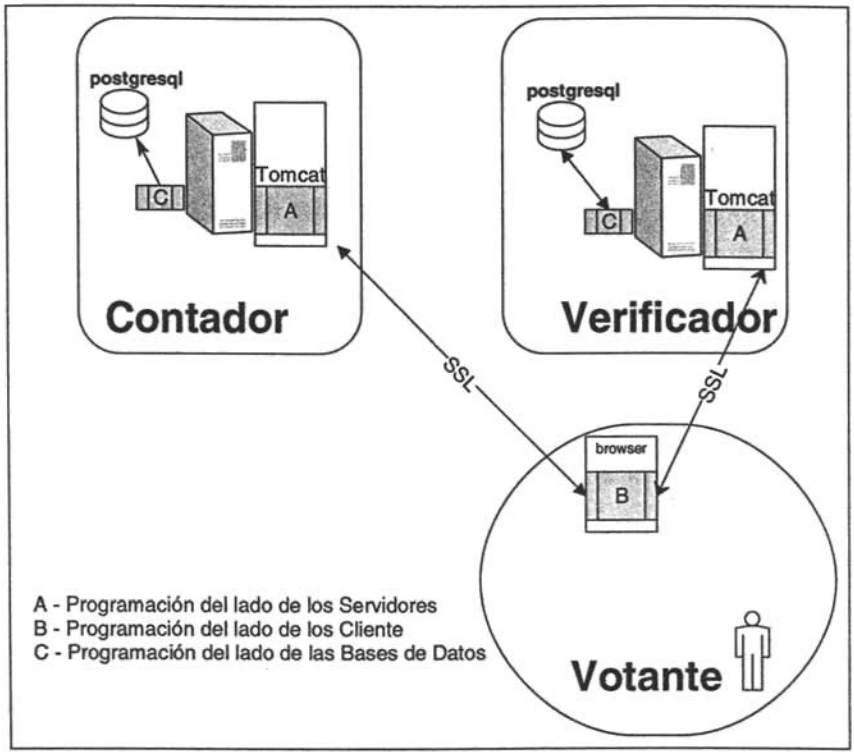


Diagrama 2

Como podemos observar en el Diagrama 2 las entidades Verificador y Contador definidas en el protocolo explicado previamente, se implementarán en máquinas diferentes, cada uno se comunicará con el votante a través de un servidor Web y almacenarán o verificarán según el caso la información que el protocolo define.

El cliente por su parte accederá vía web al Sistema de Votaciones y se comunicará con las entidades por medio de un browser.

Para poder realizar el tratamiento criptográfico del voto del usuario, se hará uso de un cliente que se integre a su browser desde el momento en el que se autentique como un usuario válido.

Para este proyecto se determinó utilizar Java como el lenguaje de programación en el que se desarrollarían los módulos de programación requeridos para las tres diferentes capas. Java además de tener capacidad para ello, y ser uno de los lenguajes más populares para este tipo de desarrollos, tiene características que le hacen ser seguro a comparación de otros lenguajes. La manera en que el paradigma de su funcionamiento cubre la mayoría de los problemas de seguridad que las aplicaciones de éste tipo tienen se revisará a continuación.

Seguridad del Lenguaje Java

Surgido en los laboratorios de Sun Microsystems, el lenguaje de programación Java, fue diseñado originalmente para utilizarse de forma preconstruida en dispositivos electrónicos. Actualmente, se ha consolidado como un lenguaje de programación orientado a objetos de propósito general. Está diseñado específicamente para ser independiente entre plataformas, por lo cual los desarrolladores pueden escribir un programa una vez y correrlo de forma segura en cualquier tipo de plataforma. La sintaxis es similar a la de lenguajes como C y C++, pero es diferente, omite algunos aspectos y toma algunas ideas de otros lenguajes.

Java hace uso de una máquina virtual que no asume ninguna particularidad de tecnología o plataforma para funcionar, esta se sitúa en la computadora, y como cualquier máquina virtual abstracta, se convierte en una interfaz para la utilización de los recursos de ésta última. Un programa escrito en Java es compilado a un tipo de código objeto llamado bytecode, que está definido en la Maquina Virtual de Java, esta realiza verificaciones y lo ejecuta.

La seguridad del Lenguaje se sostiene en las siguientes características:

- Java es un leguaje fuertemente tipificado. Realiza verificaciones de los rangos en variables de tipo String y en los índices de arreglos. No incluye ningún tipo de constructor inseguro, como aquellos en los que se invocan arreglos sin índice que pudieran arrojar resultados impredecibles.

- El lenguaje de programación Java no permite el uso de apuntadores; y puesto que administra su memoria de forma automática, cuenta con un colector de basura que libera localidades de memoria que ya no son utilizadas. Esto permite que el lenguaje Java evite problemas de seguridad, como aquellos presentes en C y C++ con las funciones free y delete respectivamente que liberan memoria de forma explícita, posibilitando el invadir segmentos utilizados por otra aplicación.

- Un Verificador de bytecode se asegura que únicamente código escrito en Java se ejecute. Antes de ejecutarse el bytecode, el verificador es invocado y revisa que no existan violaciones en las especificaciones del lenguaje o en el manejo de memoria; así también verifica que no haya volcados de pila o conversiones de tipos ilegales.

- Actividades en tiempo de ejecución, incluyen la carga y el ligado de clases necesarias para su ejecución. Durante ese proceso, el cargador define un espacio de

almacenamiento en memoria local a la aplicación que será utilizado para asegurarse que dicho código no altere la ejecución de otros programas de Java.

- Por último los accesos cruciales al sistema son mediados por la máquina virtual, y son verificados a fin de restringir los privilegios sobre todo de código que no es confiable. Esto aplica sobre todo a los Applets, el código móvil que se integra a los browsers.

Para mayor información sobre la arquitectura de seguridad del lenguaje de programación Java, Gong[20] y Oaks[38] son buenas referencias.

Programación del lado de los Servidores

La programación en Web se realiza fundamentalmente por medio de CGIs. Un Cgi es un programa que recibe parámetros remotamente, se ejecuta internamente en un servidor Web y proporciona los resultados al browser del cliente que lo invocó. En el área de la programación en Web existen diversos lenguajes de programación de alto nivel que permiten crear CGIs o componentes con funcionalidades similares.

La programación en Web por medio del lenguaje Java se realiza por medio de JSPs, y servlets, los cuales proporcionan funcionalidades similares. Los servlets ejecutan código Java del lado del servidor y responden a las peticiones de los browser de los clientes, a su vez permiten hacer uso de utilerías avanzadas en este campo como el manejo de cookies y de sesiones.

Entre las Ventajas que se encuentran utilizando Servlets en vez de CGI's con otros lenguajes de programación podemos listar las siguientes:

- Eficiencia: Con los CGI's tradicionales, un nuevo proceso es iniciado en cada petición de http. La sobrecarga de generar procesos, podría abrumar el servidor. Con servlets, la máquina virtual de Java se mantiene en ejecución y manejando cada una de las peticiones como un thread y no como un proceso del Sistema Operativo. De esta forma si existen N peticiones al mismo programa, el código del CGI es cargado en memoria N veces. Con los servlets aún cuando habrá N threads en memoria, únicamente habrá una copia de la clase cargada. Esta aproximación reduce los requerimientos de memoria y ahorra tiempo al tener que instanciar menos objetos. Finalmente cuando un CGI termina de manejar una petición, el programa termina. Esta aproximación hace difícil tener en memoria caché datos reutilizables, mantener conexiones a bases de datos abiertas, y realizar otros tipos de optimizaciones sobre datos persistentes. Los servlets se mantienen en memoria después de completar una respuesta, por lo tanto es más factible almacenar datos arbitrarios entre las peticiones de los clientes.

- Portabilidad: Los Servlets al estar escritos en Java permiten ejecutarse en múltiples plataformas y servidores Web sin cambios en los ejecutables.

- Seguridad: El paradigma de seguridad que utiliza Java, evita que los servlets sean vulnerables a enviar comandos al sistema operativo o que sean explotados a través de ataques de buffer overflow. Incluso si un servlet realiza una llamada al sistema operativo (Runtime.exec) no lo realiza a través de un Shell.

Para mayor información sobre los servlets ver Hall[21]

Programación del lado de los Clientes

Como se mencionó con anterioridad, es necesario desarrollar un programa cliente, que implemente los detalles criptográficos necesarios que el protocolo de votaciones expuesto le requiere a la entidad Votante.

Este requerimiento se satisface por medio de un Applet de Java. Éste es un programa que se descarga remotamente de un sitio Web y se ejecuta localmente en el browser del cliente que acceso al sitio.

Los Applets de Java gozan de los aspectos de seguridad que se ha expuesto con anterioridad, sin embargo, aunque Java facilita la programación segura, el que el código no sea vulnerable, no quiere decir que éste no sea malicioso.

Para proveer seguridad a este tipo de programas que son descargados remotamente, resulta necesario limitar lo que puedan realizar.

Java utiliza varias técnicas para limitar lo que un programa de esta naturaleza puede realizar. Esto se obtiene por medio de un "sandbox", la clase SecurityManager, el Cargador y el verificador de Bytecode.

Sandbox

A los programas de java se les prohíbe manipular directamente el hardware de una computadora o realizar llamadas al Sistema Operativo. En vez de ello, los programas de Java se ejecutan en una computadora virtual dentro de un espacio virtual restringido, a este espacio es al que se le denomina Sandbox.

SecurityManager

Debido a que si los programas de Java estuvieran completamente aislados de su entorno perderían mucha funcionalidad, la clase SecurityManager gestiona el acceso a los recursos que el Sandbox le restringe. Ésta clase es la que determina si una operación se permitirá o no.

Cargador

Al cargar la clase, el programa encargado de esa tarea, ejecuta una verificación a fin de que ésta no desactive la seguridad proporcionado por la clase SecurityManager y el Sandbox.

Verificador de Bytecode

Aplican los controles de seguridad explicados previamente.

Programación del lado de las Bases de Datos

Debido a que la funcionalidad de los servlets estará ligada a la información que se almacenará en las Bases de datos; el código que permite realizar las consultas y almacenamiento de información en ellas, es implementado a través de los mismos servlets. En estos se importan las bibliotecas necesarias para poder realizar las operaciones de Bases de Datos que se requieran.

Los componentes de Java que permiten la conectividad con las Bases de datos son conocidos como tecnología JDBC (Java Data Base Connectivity). Tiene la ventaja de proveer una interfaz independiente de la tecnología de Base de Datos que se utilice, por lo que si ésta es cambiada posteriormente, únicamente se afectará al código que invoca a los controladores de dicha Base de Datos.

La Seguridad en las conexiones que JDBC provee depende exclusivamente de la capacidad del Manejador de Bases de Datos al que se esté realizando la conexión. En el caso de la implementación que se propone, al utilizarse PostgreSQL existe la posibilidad de cifrar la información entre el servlet y la Base de Datos. Sin embargo, debido a que en el sistema implementado PostgreSQL reside en el mismo servidor que Tomcat, esta funcionalidad no se utilizó por no resultar indispensable.

2.3 Funciones Criptográficas

La implementación de las primitivas básicas de la criptografía de curvas elípticas, no es un campo nuevo, existen diversas bibliotecas algunas públicas y otras comerciales que implementan esta tecnología en diversos lenguajes de programación en la que Java no es excepción. Por ejemplo, en versión comercial y académica se encuentra la biblioteca del IAK (Institute for Applied Information) de Austria disponibles en: http://www.iaik.tu-graz.ac.at. Otro sitio electrónico de interés es el de "Legion of the Bouncy Castle" ahí se encuentran disponibles diversas implementaciones de primitivas de criptografía de llave pública tanto de sistemas basados en RSA como de sistemas de curvas elípticas. Ver: http://www.bouncycastle.org/

JBorzoi

Para este proyecto se hizo uso de JBorzoi una biblioteca criptográfica de software libre creada por Dragongate Technologies. El hecho de que su código sea disponible para su revisión, al igual que con otros sistemas de esta naturaleza enviste a la implementación de cierta confianza que una implementación criptográfica de código propietario no tendría.

Esta biblioteca ha sido recomendada por su estabilidad, su simplicidad y el bajo riesgo de problemas de seguridad en Freshmeat un directorio de tecnologías de software libre.

Entre otros antecedentes a esta biblioteca criptográfica, se encuentra el trabajo de Infraestructura de llave pública llevado a cabo en la División de Ingeniería y Ciencias Aplicadas de la Universidad de Harvard por David J. Malan, ello debido a que en el diseño de JBorzoi se basó la implementación de dicho trabajo.

JBorzoi opera sobre curvas elípticas binarias definidas en el FIPS 186-2. Provee funciones para realizar las siguientes primitivas de criptografía:

- Generación de llaves pública y privada.
- Aritmética de puntos en una curva elíptica
- Cifrado ECIES (Elliptic Curve Integrated Encryption Scheme),
- Acuerdo de llave ECKAS-DH1 (Diffie Hellman Key Agreement)
- Firmas Digitales utilizando ECDSA (Elliptic Curve Digital Signature Algorithm).
- Cifrado AES, Generador de llave KDF2, MAC1 (Message Authentication Code)

Para este sistema no se utilizaron todas las funcionalidades que provee esta biblioteca; únicamente se hicieron uso de aquellas clases para generación de llaves, aritmética de puntos, verificación y generación de firmas digitales basadas en ECDSA.

Para una referencia más detallada sobre la funcionalidad que JBorzoi proporciona se puede revisar la documentación oficial disponible en su sitio: http://dragongate-technologies.com

Otras Funciones Criptográficas

Entre los elementos criptográficos que no son cubiertos por la biblioteca JBorzoi, se encuentra la generación de firmas digitales ciegas; para contar con esta funcionalidad se implemento una clase que realizara dicha funcionalidad realizando modificaciones a la

clase para generación y verificación de firmas digitales basadas en ECDSA que la biblioteca proporciona.

Para la realización de funciones hash, se utilizaron las clases que vienen en los componentes criptográficos que acompañan la distribución de Java 2 JSDK 1.5. Éstas clases permiten realizar funciones hash basadas en SHA, SHA1, SHA256, SHA384 y SHA512.

El Algoritmo generador de números aleatorios, a su vez también es el que viene disponible con la distribución de Java que se utilizó para la implementación de éste trabajo, propiamente es la clase SecureRandom (java.security.SecureRandom), esta clase provee números pseudoaleatorios criptográficamente fuertes, que cumplen con las especificaciones del FIPS 140-2 y las recomendaciones del RFC 1750.

2.4 Estructuras de Datos Intercambiadas en el Sistema de Votaciones.

Con fines de estructurar y proveer compatibilidad al sistema por si en un futuro se cambia alguno de los componentes del sistema propuesto por otro desarrollado en una tecnología diferente; se proporciona la sintaxis general de las estructuras de datos intercambiadas durante la ejecución del protocolo por medio de la notación ASN.1

Para la implementación de este protocolo identificamos tres estructuras básicas que son enviadas entre las entidades participantes una vez que el votante ha descargado el cliente (applet) para ejercer su voto:

1. *BlindVotoh*, esta estructura es utilizada por el votante para la obtención de la firma digital ciega, en ella envía el hash del voto calculado a partir de la concatenación de dos números aleatorios y la operación pertinente para su enmascarado.

2. *BlindDsignature*, en esta estructura el Verificador responde enviando una firma digital ciega al votante que envío el hash del voto enmascarado.

3. *Votosigned*, esta es la última estructura que se intercambia. En ella el votante ejerce su voto enviando este en texto plano, junto con los dos números aleatorios que formaron el hash que el Verificador firmó.

La sintaxis de dichas estructuras se describe a continuación.

La estructura *BlindVotoh*, es una cadena de texto, en la que los primeros caracteres hacen referencia a la versión del sistema de votaciones que se está manejando, los siguientes identifican el tipo de curva cuyos parámetros definirán la firma digital y por último se envía el hash del voto.

```
BlindVotoh ::= SEQUENCE {
 version Version,
 curve Ecurve,
 vhash Vhash }

Version::= PRINTABLESTRING (SIZE(4))
Ecurve::= PRINTABLESTRING (SIZE (10))
Vhash::= OCTET STRING (SIZE (71))
```

La estructura *BlindDsignature*, es a su vez una cadena de texto, que comparte los primeros dos campos de la estructura anterior, pero se diferencia en que esta hace llegar la firma digital ciega del hash que fue proporcionado en el paso anterior.

```
BlindDsignature::= SEQUENCE {
 version Version,
 curve Ecurve,
 vsigned Vsigned }


Vsigned::= SEQUENCE {
C  Fdcomponent,
D Fdcomponent }

Fdcomponent::= OCTET STRING (SIZE(75))
```

La estructura *Votosigned*, comparte los dos primeros campos, y en los siguientes envía el voto en texto plano, los dos números aleatorios con los que se computó el hash del voto y la firma digital.

```
Votosigned ::= SEQUENCE {
 version Version,
 curve Ecurve,
 votoplain Votoplain,
 al1  Aleatory,
 al2  Aleatory,
 vsigned Vsigned
 }

Aleatory::= OCTET STRING (SIZE (31))
Votoplain::=OCTET STRING (SIZE(90))
```

Como se puede observar, el campo vsigned, es utilizado tanto en la firma digital ciega como en la normal, eso es porque estructuralmente ambas tienen el mismo formato y longitud.

Toda la información intercambiada está en formato de texto. Ello permite que si en un futuro alguno de los componentes se cambia, el nuevo componente que se integre sólo tendrá que comunicarse usando dicho formato, ello deberá ser una tarea trivial de realizar independientemente del lenguaje de programación o tecnología que se utilice.

Capítulo 3 Implementación del Sistema de Votaciones

En este capítulo se comenzará con una explicación del funcionamiento general del Sistema de Votaciones en la que se introducirán brevemente los elementos creados para la implementación de éste; con el auxilio de las pantallas del sistema de votaciones, se transmite a su vez la visión que tendrá el usuario final de este sistema. Posteriormente se abordará con detalle la estructura y el código de cada uno de los elementos que brindan la funcionalidad a éste sistema

3.1 Funcionamiento General del Sistema de Votaciones.

Con los componentes señalados en el capítulo anterior, finalmente se puede describir el funcionamiento general del sistema de Votaciones propuesto. Para ello, a continuación se explicará la secuencia en la que interactúan los diferentes elementos del Sistema de Votaciones desde que el votante se registra como elector válido, hasta la emisión de su voto.

Secuencia de pasos para Votar Utilizando el Sistema Propuesto

i.) El paso previo a la utilización del sistema de votaciones lo realiza el votante utilizando un browser para dirigirse al sitio en donde se encuentra el sistema de Votaciones. Hecho esto, el servidor Web donde radica la entidad Verificador responde mostrando una página de Bienvenida. Ver Fig. 1:

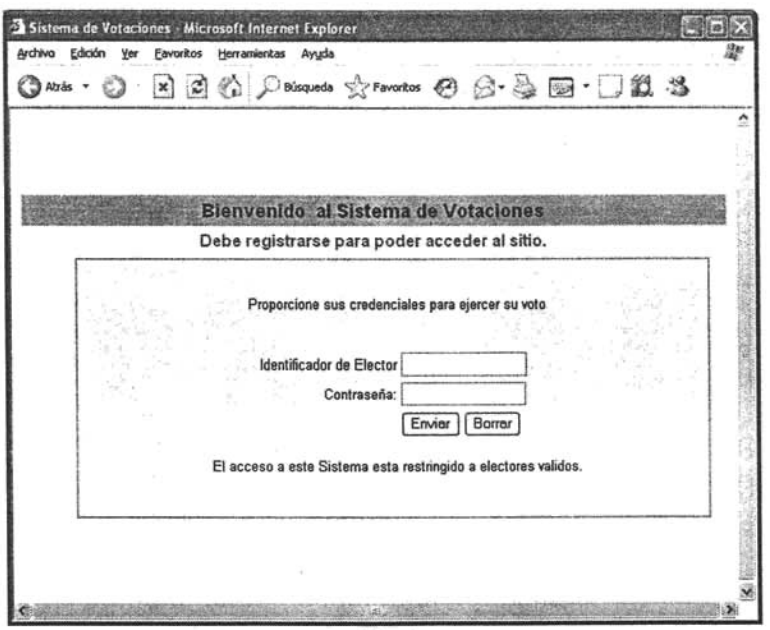


Fig. 1

ii.) Por medio de la página de bienvenida el Votante realiza su autenticación, proporcionando su número de elector y contraseña a la entidad Verificador. Estos datos son enviados vía SSL al servlet "Validador" quien por medio de consultas a la base de datos, compara las credenciales suministradas por el votante contra el registro que se tiene. De ser correctas y de no tener registrado que ese elector ya votó, se le permite al votante continuar con el proceso para emitir su voto.

Autenticado el usuario, se descarga el Applet "Voto", la interfaz por la cual el Votante ejercerá su voto, además éste ya trae parámetros de inicialización necesarios para la generación de la firma digital ciega (parámetro R y llave pública de la entidad Verificador).

Por el lado del servidor, se crea una sesión para llevar el seguimiento de los datos necesarios para que el votante puede realizar la firma digital ciega y además se define una variable en el servidor Web de alcance global con el identificador del votante a fin de registrar ese usuario en el sitio y evitar que se utilicen sus credenciales para abrir una nueva sesión y votar de nuevo; ello se realiza porque aún no se marca en la base de datos que el elector ya votó. Ver Fig. 2

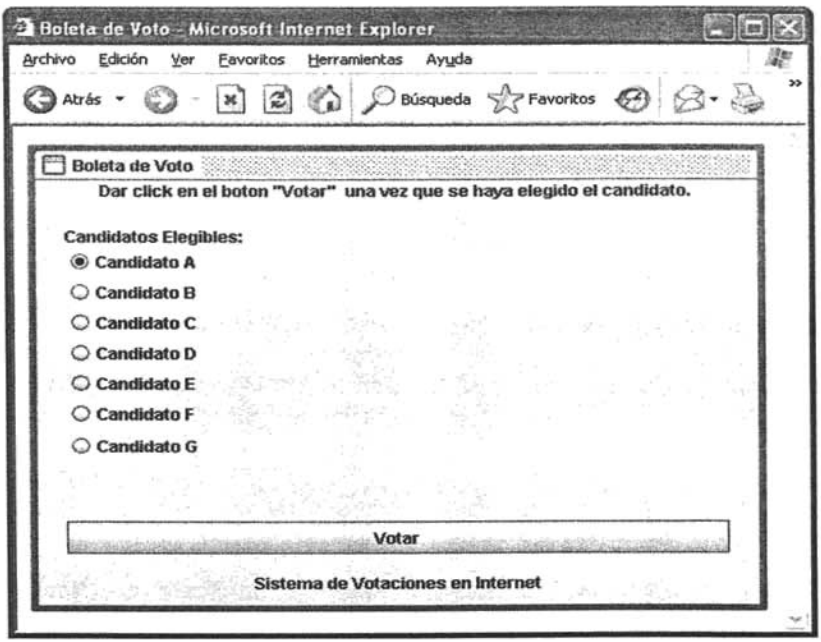


Fig. 2

iii) Cuando el Votante Elige un candidato de la lista que la interfaz le proporciona y Vota por él dando clic en el botón Votar, internamente se genera la estructura *BlindVotoh* y se envía al servlet "Firmador" alojado en el equipo del Verificador, el servlet responde enviando la estructura *BlindDsignature*. Al recibirlo el Applet desenmascara la Firma y genera la estructura *Votosigned,* la cual es enviada al servlet "Contador" que se encuentra en el equipo donde esta la entidad Contador. Éste verifica que sea valido el voto y de ser así lo almacena.

Cuando todas las acciones anteriores que corresponden al paso 2, 3, 4 y 5 del protocolo de Votaciones que se utiliza (ver capítulo 3), son finalizadas correctamente, se envía el mensaje al usuario de que su voto fue ejercido correctamente, Ver Fig. 3

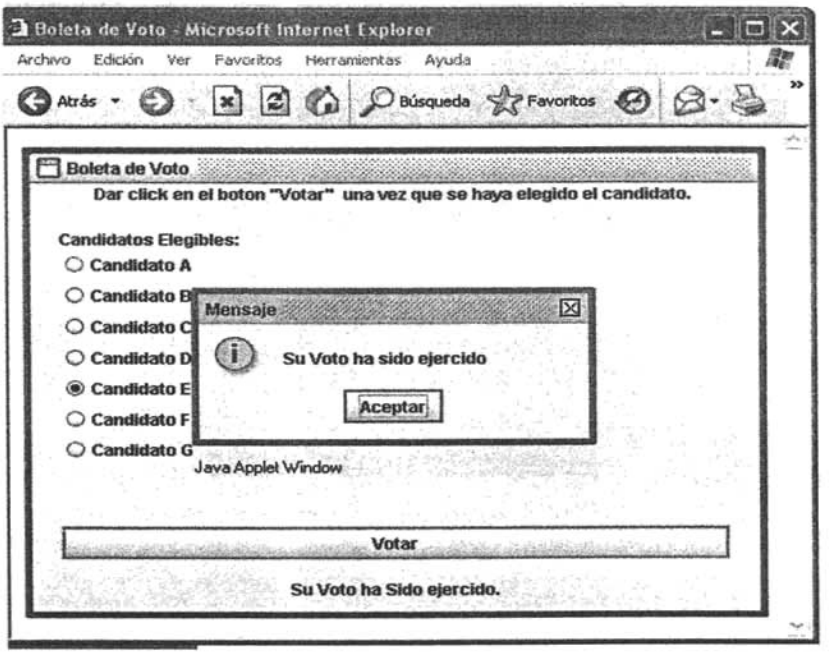


Fig. 3

La siguiente pantalla le muestra al Votante, información sobre su voto, esta información se refiere al mapeo entre la Firma Digital Ciega que quedará registrada en la entidad Verificador y la Firma Digital del Voto, que quedará en poder del Contador. Debido a que sólo el votante conoce la relación entre ambos, este es un medio para asegurarse que su voto se haya efectuado, sin verse comprometida su Identidad. Ver Fig. 4

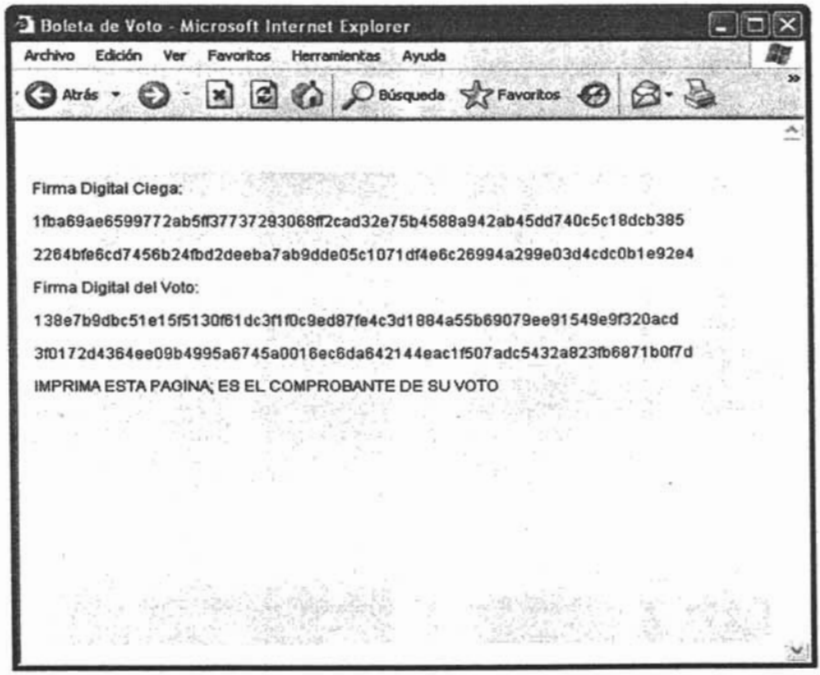


Fig. 4

La interacción de los elementos y la distribución de ellos en los equipos que emulan las entidades del protocolo de votaciones se ejemplifica en el siguiente Diagrama. Ver Diagrama 3.

DIAGRAMA DE LA IMPLEMENTACIÓN DEL SISTEMA DE VOTACIONES

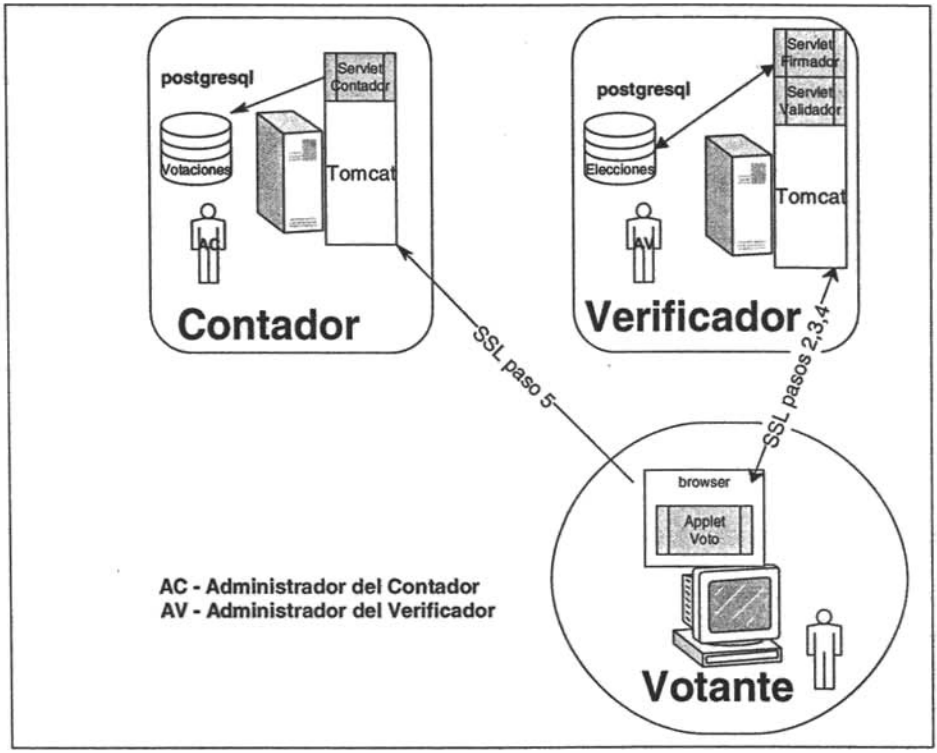


Diagrama 3

En este diagrama se puede observar los elementos que integran a cada una de las entidades, y la comunicación que realizan para permitir que el Votante ejerza su voto.

En los apartados siguientes, se revisará detalladamente la estructura y en su caso el código de cada uno de los componentes que integran a este sistema.

3.2 Bases de Datos

3.2.1 Base de Datos del Verificador

Del lado del verificador identificamos una Base de Datos, llamada electores, la cual contiene una sola tabla en la que se almacenará la lista de los votantes válidos junto con sus credenciales.

La tabla que se encuentra en la Base de datos Electores, se llama votantes. En ella se encuentran los siguientes campos:

- **id,** Éste es el identificador del votante, es una combinación simple de números y letras derivadas del nombre del votante; y forma parte de las credenciales que el votante enviará. Además este campo es la llave primaria de la tabla.
- **password,** este campo almacenará el hash del password.
- **nombre,** este campo almacena el nombre del Votante.
- **voto_firmac,** este campo almacena la firma digital ciega generada por el servlet Firmador.
- **password_plano,** este campo almacena el password en texto en claro que proporcionó el votante.

El Diagrama Físico de dicha tabla es de la siguiente forma:

Votantes
PK id char(10) NOT NULL password chkpass NOT NULL nombre varchar(50) NOT NULL voto_firmac varchar(165) Password_plano varchar(20)

A su vez se identifican dos usuarios en esa base de datos:

1. usuario *firmante*, dueño de la tabla con permisos de inserción en ella.
2. usuario *autenticador*, miembro de un grupo llamado *auten* el cual cuenta con permisos de sólo lectura sobre la tabla.

La idea de crear dos usuarios sobre esa tabla obedece a dar únicamente los privilegios necesarios a los servlets, quienes serán los que harán uso de estos usuarios para realizar las operaciones que requieran en la Base de Datos.

La sintaxis para crear la tabla es de la siguiente forma:

```
CREATE TABLE votantes
(
 id char(10) NOT NULL,
 password chkpass NOT NULL,
 nombre varchar(50) NOT NULL,
 voto_firmac varchar(165),
 password_plano varchar(20),
 CONSTRAINT llave PRIMARY KEY (id)
)
WITHOUT OIDS;
ALTER TABLE votantes OWNER TO firmante;
GRANT SELECT ON TABLE votantes TO GROUP auten;
GRANT SELECT, UPDATE ON TABLE votantes TO firmante;
```

3.2.2 Base de Datos del Contador

Del lado del Contador identificamos una Base de Datos, llamada Votaciones, la cual contiene una sola tabla en la que se almacenarán los votos que emitan los votantes y la firma digital asociada a los votos.

La tabla que se encuentra en la Base de datos Electores, se llama votos. En ella se encuentran los siguientes campos:

- **al1**, este es el primer número aleatorio con el que se computa el hash del voto, éste campo se utiliza como llave primaria.
- **al2,** este es el segundo número aleatorio con el que se computa el hash del voto.
- **candidato,** este campo se refiere al voto en texto en claro por el que el votante ejerció su voto.
- **firma,** este campo se refiere a la firma digital del voto

El diagrama físico de la tabla Votos es el siguiente:

Votos
PK al1 varchar(35) NOT NULL al2 varchar(35) NOT NULL candidato varchar(50) NOT NULL firma varchar(165)

La sintaxis para crear la tabla es la siguiente:

```
CREATE TABLE votos
(
 al1 varchar(35) NOT NULL,
 al2 varchar(35) NOT NULL,
 candidato varchar(50) NOT NULL,
 firma varchar(165) NOT NULL,
 CONSTRAINT llave PRIMARY KEY (al1)
)
WITHOUT OIDS TABLESPACE base_votaciones;
ALTER TABLE votos OWNER TO escrutador;
GRANT INSERT ON TABLE votos TO escrutador;
```

En esta base de datos identificamos al usuario *escrutador*, este usuario puede realizar inserción sobre la base de datos.

3.3 Elementos de los Servidores Web

En este apartado se revisarán los servlets, la página estática html y el applet que integra al sistema. A fin de hacer más simple su explicación, sólo se revisarán fragmentos importantes del código de ellos. Para una referencia completa del código ver el apéndice A, donde se lista el código fuente completo del sistema.

Durante la explicación del código de los servlets, se hace referencia al almacenamiento de información en bitácoras. El sistema de Bitácoras que se utiliza es proporcionado por la clase log4j, por medio de instrucciones del tipo: *logger.warn("mensaje");* instrucción que almacena en un archivo de texto plano el mensaje indicado además de otros datos especificados al momento de la inicialización del objeto *logger*.

3.3.1 Elementos del Servidor Web del Verificador

En este apartado se definirán los elementos que existen dentro del servidor Web del Verificador. Para ello definimos $TOMCAT_HOME como la variable que identifica a toda la ruta del sistema de archivos donde se encuentra alojado el servidor Web.

$TOMCAT_HOME/webapps/Root, es el directorio que publica por omisión el servidor Web Tomcat.

En ese directorio únicamente se encuentra el archivo **index.html**. Éste archivo tiene una tabla en html que contiene la forma de autenticación que utiliza el Votante para proporcionar su identificador de votante y contraseña a fin de ser autenticado y poder de esta forma descargar el applet que le permita ejercer su voto.

La siguiente línea es la que utiliza index.html para enviar al servlet Verificador el identificador de votante y la contraseña por SSL:

<form method=post action="https://host-verificador:8443/votaciones/Verificador">

$TOMCAT/votaciones/WEB-INF/classes/com/elliptic, es la carpeta que contiene los servlets que contestarán a las peticiones de la página de autenticación y a las de el Applet Voto una vez que éste se haya descargado.

Servlet Validador

En **$TOMCAT/votaciones/WEB-INF/classes/com/elliptic** se ubica el servlet **Validador.class,** éste recibe vía SSL el identificador del votante y el password del votante.

Las líneas trascendentes de éste servlet se explican a continuación:

En el siguiente fragmento de código se muestra el método que se ejecuta cuando los datos llegan vía Post, en caso de que lleguen vía Get se manda un mensaje de error.

```
public void doPost(HttpServletRequest request, HttpServletResponse res)
throws IOException, ServletException
{
```

Se guardan los datos que fueron enviados a variables de tipo String. Posteriormente se realiza una serie de verificaciones en las que compara que el tamaño de los parámetros sea el esperado, y que tengan el tipo de caracteres se espera. En caso de que no sea así se envía un mensaje de error al browser del cliente, y se registra el evento en una bitácora del servidor Web. En el siguiente fragmento de código únicamente se muestra la validación hecha para la longitud de los parámetros.

```
String id = request.getParameter("id_votante");
String password = request.getParameter("password");
int largo = request.getContentLength();
if(largo>60)
{
  out.println("<html>Error las credenciales proporcionadas son incorrectas");
  out.println("<br> <a href=http://localhost:8080/> regresar </a></html>");
  logger.warn("Envio de Credenciales exceden longitud valida - "+ipaddress);
}
```

Después de cargarse los drivers de PostgreSQL y de realizarse la conexión a la base de datos con el usuario autenticador, se realiza una consulta a la tabla Votantes para verificar que el identificador del votante exista, de ser así, se selecciona el *id, password y voto_firmac*:

```
Statement st = conn.createStatement();
ResultSet rs = st.executeQuery("SELECT id,password," +
        "voto_firmac FROM elecciones.votantes WHERE id= " + id );
```

Si el identificador del votante no existe, se envía un mensaje de error al browser del cliente, y se registra el evento en una bitácora del servidor Web.

Si la consulta devuelve resultados, el identificador de votante si existe, se guardan los datos que se solicitaron en la consulta y se compara el hash del password proporcionado por el votante con el hash almacenado en la base de datos:

```
if (rs.next())
{
 id_q=rs.getString(1);
 password_q=rs.getString(2);
 voto_firmac_q=rs.getString(3);
 rs = st.executeQuery("SELECT chkpass_out(\'" + password + "\')");
 rs.next();
 if (password_q.equals(rs.getString(1)))
```

En caso de que no sea igual el hash almacenado a aquél que se calculó, se envía un mensaje de error al browser del cliente, y se registra el evento en una bitácora del servidor Web

Si la contraseña es correcta, se realiza una verificación posterior para verificar que el valor del *voto_firmac* sea nulo demostrando que aún no ha ejercido su voto el Votante. Además se debe verificar que el usuario no haya abierto una sesión antes, este último dato se almacena en la variable *listado* en el campo *id:*

```
if (voto_firmac_q==null)
{
    rs.close();
    st.close();
    conn.close();
    if(listado.getAttribute(id)!=null)
```

En caso de que el votante no haya votado aún, se preparan las instrucciones para enviar el Applet del Voto, la llave pública de la entidad Verificador y el punto Aleatorio R para la realización de la firma ciega. En caso contrario, se envía un mensaje de error al browser del cliente, y se registra el evento en una bitácora del servidor Web

En las siguientes instrucciones, se calcula un punto aleatorio R = kQ, con el tipo de curva de campo finito binario de 283 bits, curva que se utilizó para la implementación de éste sistema.

```
ECDomainParameters dp = ECDomainParameters.NIST_B_283();
ECPrivKey KBlindc= new ECPrivKey(dp);
ECPubKey RBlindc = new ECPubKey(KBlindc);
```

Posteriormente, se registra en una variable de ámbito global al servidor Web, el identificador del elector a fin de que no pueda volver a usar esas credenciales. Además se prepara el sistema para crear una sesión del usuario autenticado; se añade a los atributos de la sesión el id del elector y el punto R calculado:

```
listado.setAttribute(id, "registrado");
HttpSession session = request.getSession(true);
session.setAttribute("KBlindc",KBlindc);
session.setAttribute("RBlindc",RBlindc);
session.setAttribute("elector",id);
```

Por último se redirecciona al votante con el Applet que será su cliente para emitir su voto. Proporcionándole su número de sesión a fin de que pueda autenticarse desde el Applet junto con parámetros del punto aleatorio R (RBlindc) y la llave pública del Verificador (PK):

```
out.println("<applet code=com.elliptic.Voto.class archive=sVoto.jar
            width=\"100%\" height=\"100%\">");

out.println("<param name=\"sessionId\" value="+
            session.getId()+">");
out.println("<param name=\"RBlindcx\" value="+
            RBlindc.W.x.val.toString()+">");
out.println("<param name=\"RBlindcy\" value="+
            RBlindc.W.y.val.toString()+">");
out.println("<param name=\"PKx\" value="+ px+">");
out.println("<param name=\"PKy\" value="+ py+">");
```

Como se puede observar se proporciona el parámetro archive =sVoto.jar, este es la firma digital sobre el Applet utilizando el certificado digital con el que está funcionando SSL; esto se realiza para proveerle los privilegios suficientes al applet para que pueda realizar conexiones tanto a la entidad Verificador como a la entidad Contador.

Servlet Firmador

Situado también en **$TOMCAT/votaciones/WEB-INF/classes/com/elliptic,** el servlet Firmador recibe la petición de la firma digital ciega una vez que el Votante ha elegido un candidato en el Applet del Voto.

Primeramente verifica que el Cliente que le solicita la firma digital se encuentre autenticado:

```
HttpSession session = request.getSession(false);
if (session != null)
```

De no ser así, se envía un mensaje de error al applet y se registra el error en una bitácora.

Si el usuario si inició sesión, después de hacer verificaciones de longitud sobre la estructura *BlindVotoh*, se obtiene el hash del voto que fue enviado, convirtiéndolo de String al tipo BigInteger. A su vez se prepara la llave privada para utilizarla para firmar el voto.

```
Voto = new BigInteger(sVoto,16);
llave = new BigInteger(key,16);
```

En las siguientes instrucciones se instancia un objeto de tipo llave privada ECPrivKey, y un objeto de la clase BECDSA, para realizar la firma digital ciega. Para ello se invoca el método initSignature con la llave secreta, el parámetro K del punto aleatorio R, el punto R y el hash del voto:

```
ECPrivKey sk = new ECPrivKey(dp, llave);
BECDSA VCF = new BECDSA();
VCF.initSignature(sk,KBlindc,RBlindc,Voto.toByteArray());
```

Se realiza la firma digital, invocando el método sign(), y se escribe en la Base de datos la firma digital ciega en el registro del votante, para indicar que este ya votó:

```
VCF.sign();
int result = st.executeUpdate("UPDATE elecciones.votantes "+
"SET voto_firmac = \'"+VCF.c.toString(16)+" "+VCF.d.toString(16)+
"\' WHERE id = "+elector);
```

Finalmente se envía el objeto BlindDsignature al cliente.

```
OutputStream outstr = res.getOutputStream();
ObjectOutputStream oos = new ObjectOutputStream(outstr);
oos.writeObject(BlindDsignature);
oos.flush();
oos.close();
```

Applet Voto

Ubicado en **$TOMCAT_HOME/webapps/Votaciones/com/elliptic/** , encontramos el Applet llamado **Voto** que servirá como programa cliente para que el Votante pueda ejercer su voto.

Este Applet utiliza Bibliotecas de Java swing para ofrecer la interfaz gráfica en la que el votante pueda elegir de una lista de opciones el cómo votará. Cuando lo ha hecho, se transfiere el control a un método llamado votar(), que será el que se describirá.

Una vez leídos los parámetros proporcionados por el Verificador, que son la llave pública de esa entidad y el punto aleatorio R, al cual se le almacena en la variable RBlindc; se realizan los pasos necesarios para preparar una firma digital ciega.

Se calcula un nuevo punto como la multiplicación de un número aleatorio alfa por R, a este nuevo punto se le llama RBlind

```
RBlind = (ECPointF2m)RBlindc.dp.E.mul(alfa,RBlindc.W);
```

Hecho lo anterior se calcula el componente beta de la máscara como la parte X del punto Rblindc, por el inverso de RBlind por alfa.

```
XRinv = Utils.OS2IP(Utils.FE2OSP(RBlind.x));
XRinv = XRinv.modInverse(dp.r);
beta = Utils.OS2IP(Utils.FE2OSP(RBlindc.W.x)).mod(dp.r);
beta = beta.multiply(XRinv).mod(dp.r);
beta = beta.multiply(alfa).mod(dp.r);
//aqui Beta ya fue calculado como la parte X de RBlindc por
//el inverso de la parte X de RBlind por alfa
```

Se calcula el hash del voto dos veces utilizando los dos números aleatorios. Cabe mencionar que La clase Utils, definida en la Biblioteca de JBorzoi permite realizar inversiones y conversiones en los elementos de un arreglo; aquí se utiliza para convertir un byte[] a String:

```
sha = MessageDigest.getInstance("SHA-256");
M = M.concat(AL1.toString());
sha.update(M.getBytes());
M0 = sha.digest();
```

```
M=Utils.intArrayToString(Utils.toIntArray(M0));
M=M.concat(AL2.toString());
sha.update(M.getBytes());
M0 = sha.digest();
```

Se aplica la máscara al hash del voto, ahora almacenado en M1 de tipo BigInteger como la multiplicación del hash del voto por la variable beta y por el inverso de alfa.

```
M1=Utils.OS2IP(Utils.revIntArray(Utils.toIntArray(M0)));
M1=M1.multiply(beta).mod(dp.r);
M1=M1.multiply(alfa.modInverse(dp.r)).mod(dp.r);
```

Una vez listo el hash del voto, se envía al Firmador el objeto *BlindVotoh* quien responderá con *BlindDsignature* que será la firma digital asociada al hash del voto enmascarado que se envió.

Después de verificar que la firma digital ciega sea correcta, se elimina la máscara de ésta a fin de tener una firma digital normal. Mf2 es la firma digital que se obtuvo.

```
Mf2.c=Utils.OS2IP(Utils.FE2OSP(RBlind.x)).mod(dp.r);
Mf2.d=Mf2.d.multiply(beta.modInverse(dp.r));
Mf2.d=Mf2.d.mod(dp.r);
```

Realizado lo anterior, se prepara la estructura *Votosigned* y se envía al contador.

Una vez enviado el *Votosigned* al contador, se le presenta en el Applet al usuario la firma digital ciega y la firma digital, sugiriendo su impresión a fin de poder obtener un comprobante del ejercicio de su voto.

3.3.2 Elementos del Servidor Web del Contador

En este apartado se definirán los elementos que existen dentro del servidor Web del Contador . $TOMCAT_HOME como en la sección anterior se utilizará como la variable que identifica a toda la ruta del sistema de archivos donde se encuentra alojado el servidor Web.

Servlet Contador

El Servlet contador se ubica en **$TOMCAT_HOME/webapps/Votaciones/com/elliptic,** de la máquina que contendrá a la entidad Contador.

Recibida la estructura Votosigned, se realizan verificaciones de tamaño y caracteres en la estructura *Votosigned* enviada por el Votante. Hecho esto se almacena en las variables AL1B, AL2B, M, FCB, FDB, los valores del primer y segundo número aleatorio, el voto en texto plano y los dos componentes de la firma digital respectivamente.

Posteriormente se verifica que la firma digital que fue enviada sea válida para esos parámetros preparando el hash del voto de la misma forma en la que lo procesaba el Applet del Votante:

```
M = M.concat(AL1B);
sha.update(M.getBytes());
M0 = sha.digest();
M=Utils.intArrayToString(Utils.toIntArray(M0));
M=M.concat(AL2B);
sha.update(M.getBytes());
```

Finalmente se verifica que la firma digital sea válida, para ello se instancia el objeto Mf de la Clase ECDSA con los parámetros de inicialización que corresponden a la firma digital recibida. Posteriormente con el método initVerify() se le proporciona la llave pública del Verificador al objeto, con el método update() se proporciona el hash del voto y el método verify() realiza la verificación regresando un valor booleano.

```
ECDSA Mf= new ECDSA(FCB,FDB);
Mf.initVerify(pk);
Mf.update(M.getBytes());
if(Mf.verify())
```

Si la firma digital es válida el voto se toma por bueno y se almacena en la Base de Datos, en caso contrario, se genera un registro en bitácoras.

```
st.execute("INSERT INTO elecciones.votos" +
                    " (al1,al2,candidato,firma) VALUES " +
                    "(\'" + al1 + "\'," + "\'" + al2 +
                    "\'," +
                    "\'" + candidato + "\'," + "\'" + fc +
                    "--" + fd + "\')");
```

Al almacenar en la base de Datos del Contador el voto, se da por ejercido éste.

3.4 Pruebas de Desempeño del Sistema.

Para efectos de determinar el desempeño del Sistema de Votaciones construido, se montó una maqueta con los siguientes elementos.

Equipo para la entidad Votante:

CPU: Pentium III 700 Mhz
Memoria: 320 MB RAM
Sistema Operativo Windows Me

Equipo para la Entidad Verificador:

CPU: Pentium IV a 3Ghz
Memoria: 1 GB RAM
Sistema Operativo: Windows Server 2003

Equipo para la entidad Contador:

CPU: Celeron a 433 Mhz
Memoria: 198 MB RAM
Sistema Operativo: Windows Server 2003

Con la configuración mencionada, se realizó la emisión de un voto desde el equipo del cliente varias veces. Tras votar repetidas veces, se determinó un tiempo de respuesta máximo de 50 segundos. Este tiempo se contó a partir de que el cliente da clic en enviar credenciales en la página de inicio del Sistema, hasta que le aparece su comprobante de voto después de ejercerlo. Cabe mencionar que mucho del tiempo es consumido por el Cliente de Votaciones durante la inicialización de la interfaz gráfica del Applet así como las verificaciones y operaciones que realiza con la firma digital del voto. Este tiempo puede reducirse, hasta 30 segundos si el Cliente de Votaciones no cierra el browser, lo cual ahorra tiempo durante el establecimiento del canal seguro de SSL.

A continuación se analizará con más detalle el tiempo de ejecución de las entidades Verificador y Contador auxiliándose de las métricas que brinda la utilería Windows Task Manager; la cual por medio de una cuadricula estructura los tiempos en bloques de 6 segundos.

3.4.1 Métricas de Desempeño del Verificador

A continuación, se muestra el comportamiento en cuanto a procesador y memoria del equipo que alojó a la entidad Verificador con la ayuda de la Gráfica 1.

Procesador

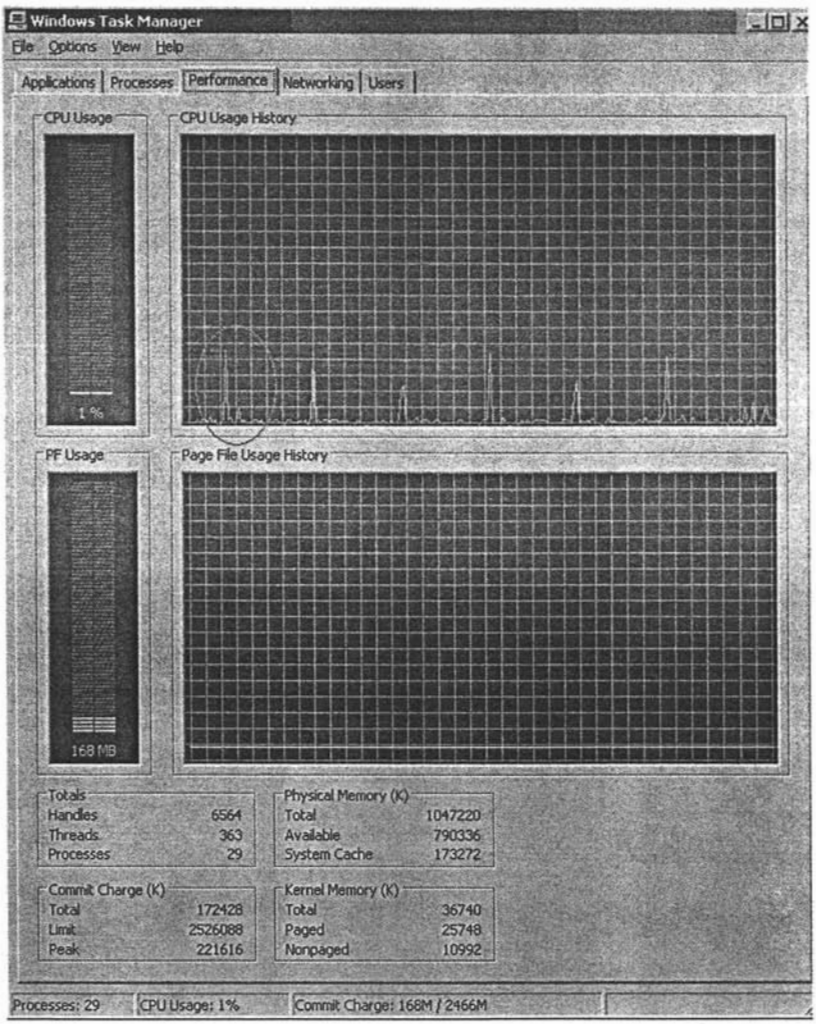
En la Gráfica 1, puede observarse en el apartado *CPU Usage History* que el procesador muestra picos de 1 segundo de hasta el 30% de utilización; Estos picos se suceden en el momento en el que un votante logra autenticarse correctamente; cabe recordar que es en ese tiempo cuando se crean varios objetos en memoria en el servidor y se calculan tanto el punto aleatorio que servirá como insumo a la firma digital ciega, como la cadena aleatoria que crea el servidor web para el manejo de sesión.

A su vez puede observarse en dicha gráfica que tras haberse autenticado el votante, el costo de la generación de la firma digital de su voto y su inserción en la base de Datos es mínimo. Ello genera un requerimiento en procesamiento de hasta 3 segundos con un 1% adicional de carga. (Ver segundo pico del elipse rojo de la figura 1)

Durante las pruebas se hicieron intentos de autenticación con credenciales erróneas y la repercusión en el procesamiento no fue perceptible.

Memoria

Se puede observar también en la gráfica 1 dentro del apartado *Page File Usage History*, que en cuanto a la Memoria, esta no se ve impactada con la emisión del voto de un solo votante.



Gráfica 1

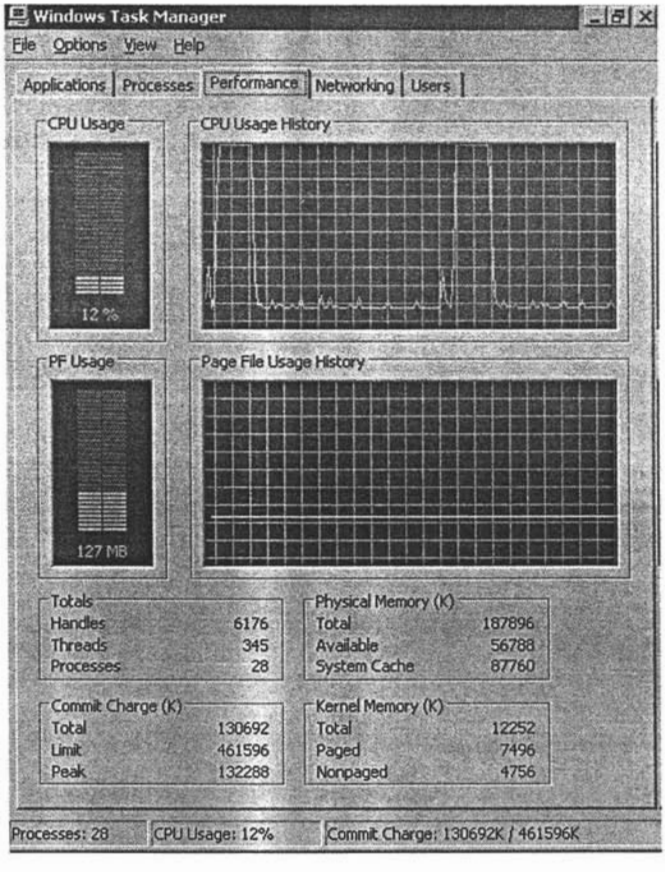
3.4.2 Métricas del Desempeño del Contador

Procesador

Para el caso del Contador, y debido por la pobre capacidad de procesamiento que este tiene la situación fue muy diferente. Como se puede ver en la gráfica 2, el costo de verificar la firma digital del voto y de insertarla en la base de datos, tuvo un costo de al menos 12 segundos con el CPU al 100%

Memoria

Como en el caso del Verificador, se puede observar en la gráfica 2 dentro del apartado *Page File Usage History,* que en cuanto a la Memoria, esta tampoco se vio impactada con la emisión del voto de un solo votante.



Grafica 2

3.4.3 Observaciones a las pruebas de rendimiento del Sistema

Las pruebas hechas solo reflejan el comportamiento del sistema con un solo votante, y los equipos tienen una arquitectura de computadoras personales y no de servidores, los cuales cuentan con discos, y procesadores más veloces. Por lo que sería conveniente, realizar pruebas de estrés a este sistema, utilizando equipos con arquitectura de servidores para obtener indicadores precisos de su capacidad y conveniencia en desempeño.

Por último, teniendo a la entidad Contador ejecutándose en un equipo similar al que se utilizó para el Verificador, por los picos encontrados en el momento de autenticarse, se podría estimar que se podrían soportar hasta 100 usuarios conectados de forma simultánea.

3.5 Consideraciones finales del Sistema de Votaciones

Como se explicó previamente en el protocolo, para que este sistema funcione se debe evitar que exista comunicación entre el Contador y el Verificador, ello a fin de que no se pueda romper la privacidad del votante a través de correlacionar la forma en la que van llegando lo votos con la manera en la que se van registrando los votantes.

Para este fin se pueden realizar las siguientes acciones.

1. En los servidores donde radican los sistemas de votaciones se debe bloquear el acceso remoto a los servidores y negar la comunicación entre el Verificador y el Contador. Esto se puede realizar por medio de firewalls integrados en los servidores que albergan a las entidades y a nivel de los equipos de comunicaciones.

   A nivel de los equipos de la red de computadoras, el bloqueo de la comunicación se puede realizar por medio de una configuración ruteador, firewall, ruteador; situados entre las subredes en las que se encuentran el servidor de la Entidad Verificador y la Entidad Contador, a fin de que el primer ruteador corte el tráfico y en caso de verse comprometido, el firewall impida que se comprometa el segundo ruteador y el acceso continúe bloqueado en todo momento. Ver Diagrama 4

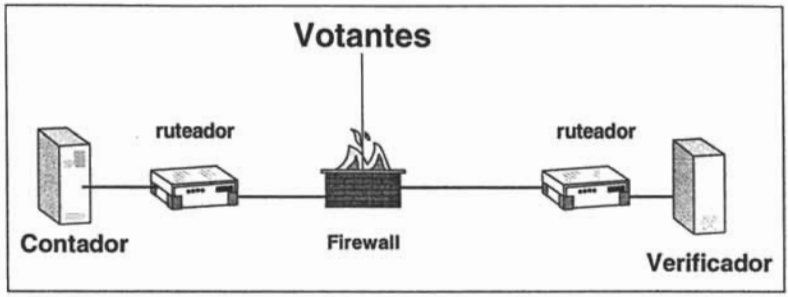


Diagrama 4

2. Implementar una buena seguridad física para el resguardo de los servidores. Y que el único acceso a los servidores sea vía consola, ello además de permitir vigilar a observadores las acciones que realicen los administradores de las entidades, es una buena práctica para eliminar a posibles atacantes.

3. Como en las elecciones tradicionales, deben existir observadores de cada una de las partes involucradas en la elección.

Un aspecto a cuidar es la posibilidad de que la Entidad Verificador genere sus propios votos al ser, responsable de la llave privada que le da validez a los votos que envían los votantes.

Como se mencionó en el protocolo esto se evita si se incluye como requisito el almacenar en la base de datos el password del votante en texto plano. Y ya que el verificador tiene en su poder únicamente el hash de los passwords, no cuenta con la suficiente información para realizar falsificación de votos sin ser detectado.

Lo anterior es válido sólo si se cuidan los siguientes aspectos:

- Los passwords de los votantes son criptográficamente fuertes, es decir que son difíciles de adivinar por ser lo suficientemente complejos. Utilizan caracteres alfanuméricos y especiales, no utilizan palabras de diccionario y tienen una longitud superior a 10 caracteres.

- Los passwords de los votantes son generados sin la participación de la entidad Verificador a quien únicamente se le proporcionaría la relación final con el hash de los passwords. Esta se realizaría por medio de una autoridad electoral ajena a las entidades del sistema, la cual generaría los passwords y se los entregaría de forma segura a los votantes y el hash de estos al Verificador.

Observaciones:

El incluir a una autoridad electoral adicional como responsable de generar los passwords, evita que las entidades electorales definidas en el Sistema tengan en su poder la información para votar en lugar de los votantes. Sin embargo, esta medida no funciona en caso de que la autoridad encargada de generar los passwords, se corrompa y proporcione esta información a las entidades del Sistema de Votaciones. Como se mencionó en el capítulo 2, no existe protocolo de Votaciones que planteé defensa efectiva contra ataques internos.

Como se verá más adelante en el Capítulo 5, la implementación de este sistema dentro en un esquema de módulos o kioscos, ayudaría a disminuir el impacto en caso de que las contraseñas se comprometieran.

Por último el problema de generar una contraseña robusta puede solucionarse si se establece como procedimiento obligatorio para generarla el utilizar alguna implementación de APG (Automated Password Generator definido en el FIPS 181). Implementaciones de esta herramienta se encuentran disponibles en versiones comerciales y gratuitas.

Capítulo 4 Limitaciones, Amenazas y Escenarios de Implantación

En este capítulo se expondrán las limitaciones y principales amenazas inherentes a los Sistemas de Votaciones por Internet; a su vez, se plantearán las alternativas de solución a los obstáculos identificados, según los diferentes escenarios en que podría implementarse el Sistema de Votaciones propuesto.

4.1 Limitaciones del Sistema de Votaciones

Como se identificó en el estudio hecho al SERVE (ver Jefferson[27]); tal vez Internet permita realizar comercio electrónico, actividad que sin ser completamente segura, ha adquirido cierta confianza por parte de los que participan en ella. Sin embargo, los sistemas de votaciones difieren de los de comercio electrónico en algunos aspectos clave que vuelven a la seguridad disponible para Internet insuficiente. Las principales diferencias son las siguientes:

1. Existe un tiempo límite; A diferencia de las transacciones de comercio electrónico, en los esquemas de votaciones existe una ventana limitada de tiempo en la que se debe ejercer el voto. Si durante ese tiempo los servicios no están disponibles, las votaciones no podrían ser válidas.

2. Anonimato; Muchos de los controles para recuperar una transacción de comercio electrónico en caso de que se vea interrumpida por una falla en los sistemas, no pueden ser aplicados para las votaciones debido al requerimiento de tener que mantener el anonimato del votante.

3. Atacantes; Por las decisiones que involucran a las elecciones, pueden encontrarse intereses aún más fuertes por querer tomar el control de ellas que las que existen en el comercio electrónico.

4.2 Amenazas al Sistema de Votaciones

La gran ventaja que Internet ofrece al sistema propuesto al permitir utilizar su infraestructura, es también su principal problema. Ello se debe a que la plataforma de Internet por sí misma genera múltiples amenazas para cualquier máquina que se conecta a ella. A continuación se mencionarán las más relevantes.

Vulnerabilidades en los equipos de Comunicaciones.

Cada elemento de la red de Internet puede convertirse en un punto de ataque. Cada DNS, Router, Firewall y Switch puede estar configurado de forma insegura, lo que podría permitirle ganar a un atacante control sobre una parte de la infraestructura a fin de utilizarla para montar un ataque de hombre en medio a fin de suplantar a las entidades de votaciones válidas.

Ataques de Negación de Servicios Distribuidos

Uno de los problemas aún abiertos, es la amenaza de ataques Distribuidos de Negación de Servicios. Un ataque de este tipo consiste en comprometer un gran número de equipos conectados en red, a los cuales se les instala un agente que permita ganar control remoto de ellos. Esto se realiza a fin de enviar peticiones masivas por medio de dicho grupo de máquinas a un objetivo el cual el atacante pretende saturar a fin de que la disponibilidad de dicha máquina se pierda.

Un ataque de esta naturaleza sobre alguna de las entidades del Sistema podría interrumpir las elecciones si no se toman los mecanismos de defensa necesarios. Aún cuando han surgido alternativas de filtrado en equipos de comunicaciones comerciales que permiten detectar y reducir la magnitud del problema; no hay una solución única ni medida que garantice protección. El problema con este tipo de ataques es que no se irán y no hay forma de prevenirlos. Se han presentado desde hace tiempo y cada vez surgen nuevas herramientas que hacen más sencillo el ejecutarlos. (ver Mirkovic [33])

Código Malicioso

Internet actualmente es un medio que permite la fácil diseminación de Virus y Gusanos sobre todo en equipos que no cuentan con la protección necesaria. Sin embargo, aún cuando la configuración de los equipos conectados a Internet cuenten con software que mitigue estas amenazas como un Antivirus, la seguridad que este pueda proveer será sobre código malicioso conocido, pero de muy poco podrá ayudar si el virus o gusano es nuevo.

En este caso si buena parte de los clientes que se utilicen para usar el Sistema de Votaciones se ven comprometidos, la seguridad del protocolo de poca ayuda será, las votaciones se habrán salido de control.

Ingeniería Social

A pesar del incremento en el uso de Internet, es relativamente poca la proporción de personas familiarizadas no sólo con esta tecnología sino sobre la utilización de una computadora. Esta falta de cultura Informática, puede generar problemas en los que algunas personas pueden tomar ventajas. Ayudar a una persona con la interfaz de votaciones en la que se emitirá el voto, puede ser una forma sencilla de robarle su voto.

Además, falsos correos advirtiendo que el sitio para realizar las votaciones ha cambiado, o que es necesario confirmar la información del elector solicitándole sus credenciales , pueden ser una forma fácil en la que un atacante puede suplantar al sistema original a fin de obtener datos confidenciales que podrían ser utilizados para robar votos. (ver Rubin[48])

Amenazas Políticas

Dependiendo de la magnitud de las elecciones, podría darse el caso de que organizaciones terroristas o gobiernos extranjeros invirtieran fuertes sumas de dinero a fin de sabotear o alterar las elecciones.

4.3 Escenarios de Implantación

Las estrategias que permitirían proveer la seguridad requerida así como mitigar las amenazas del ambiente tienen diferentes alcances según los escenarios en los cuales se implante el sistema de Votaciones. A continuación se analizarán los diferentes escenarios de implantación del Sistema con las medidas de seguridad recomendadas.

4.3.1 Seguridad para Votaciones Remotas

Seguridad para los Servidores del Sistema de Votaciones

La seguridad del lado de los Servidores del Sistema de Votaciones, se obtiene en gran medida por medio de una estructura de red de computadoras segura; esto significa que los equipos de comunicaciones a los que están conectados los servidores del Sistema deben tener una configuración que permita proteger a los equipos de posibles intrusiones y ataques.

Una de las principales prácticas para asegurar los servidores de una Red, es separarlos de Internet por medio de Firewalls y ruteadores.

A su vez por medio de esos dispositivos se deben definir zonas que permitan separar el tráfico según los servicios que se proporcionen en cada parte de la red. Esto facilita la tarea de los dispositivos de seguridad, pues se tipifica el tráfico y se tienen más elementos para identificar cuando existe una anormalidad en la red.

Para las Votaciones Remotas, el Sistema debe poder ser accesible desde cualquier parte de Internet, por lo que los equipos de comunicaciones deben permitir el tráfico desde cualquier parte; pero únicamente a las subredes donde se encuentran los Servidores que contienen a las entidades del Sistema; siempre y cuando las peticiones tengan las características que se esperan, negando cualquier tráfico con características diferentes o que busque acceso a otras subredes dentro de la infraestructura donde se implante el Sistema.

Para poder efectuar lo anterior, el Sistema de Votaciones se situaría en una red de computadoras bajo el siguiente diseño:

- Toda la red donde se implemente el sistema estaría detrás de un ruteador principal equipo que sería el enlace con Internet y quien realizaría filtrado de paquetes;
- En una segunda capa se situaría un firewall configurado a nivel de Aplicación para el análisis de los paquetes. Este firewall tendría al menos tres interfaces para separar la subred del Verificador, la del Contador, y la red Interna.
- En una tercera capa, se situaría antes de cada subred un ruteador que proporcionará defensa a profundidad sobre el tráfico que sea transmitido entre la zona de los servidores del Sistema de votaciones.

Ver Diagrama 5

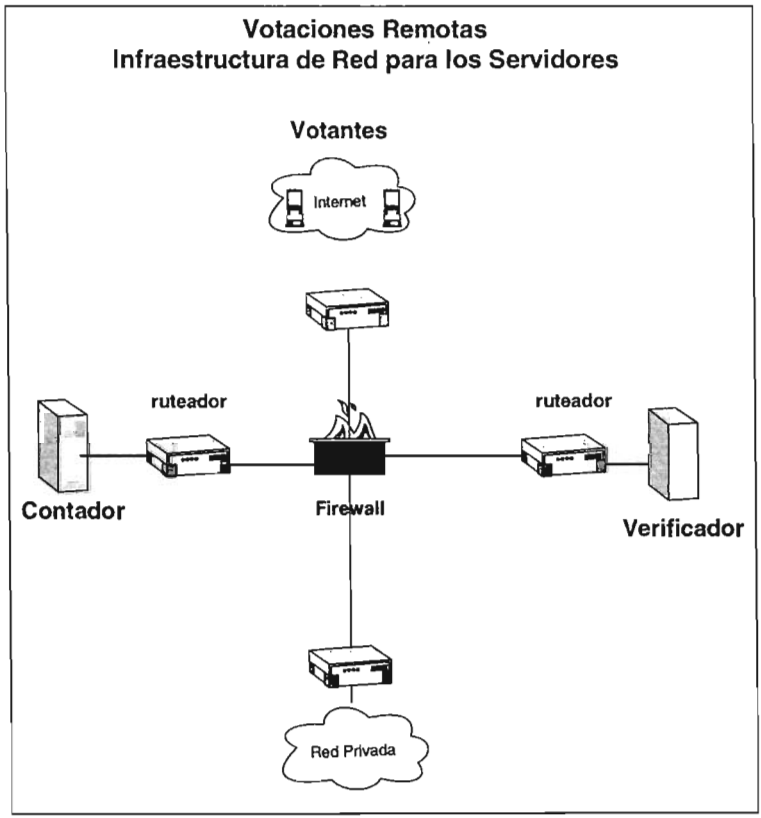


Diagrama 5

Además de la configuración mencionada en la red de computadoras los servidores del sistema de Votaciones deben contar con los siguientes componentes:

- Un Firewall instalado a nivel del servidor
- Ultimas Actualizaciones de Seguridad al Sistema Operativo.
- Un Antivirus.

Con los elementos expuestos, una configuración de esta naturaleza podría proteger de ataques con código malicioso y de posibles intrusiones a los servidores o a la red donde se implemente el sistema de votaciones. Sin embargo, para protegerse contra los Ataques de Negación de Servicios Distribuidos, se deben contemplar otros aspectos.

Defensa contra Ataque de Negación de Servicios Distribuido

Una Plan de defensa contra un ataque de Negación de Servicios idealmente debe abarcar la prevención y reacción ante una amenaza de esta naturaleza.

Prevención

Debido a que el sistema bajo el esquema de Votaciones Remotas debe ser accesible desde cualquier parte de Internet, evitar que un atacante comprometa los equipos necesarios para lanzar un ataque de Negación de Servicios Distribuido, implicaría robustecer la seguridad de las máquinas conectadas en Internet, tarea que no es viable.

Por esta razón, en un esquema de Votaciones Remotas, la prevención a un Ataque de Negación de Servicios Distribuido más bien dependerá de tener alta disponibilidad en los servidores por medio de soluciones de balanceo de carga o clusters, así como equipos de red de gran capacidad; de esta forma, en caso de sufrirse un ataque de ese tipo, se tendrán elementos para continuar ofreciendo el servicio pese a un considerable incremento en las peticiones al Sistema de Votaciones.

A su vez es recomendable de ser posible, establecer acuerdos con el *ISP*[1] que proporciona el enlace con Internet para que en caso de una contingencia, se pueda aplicar algún filtrado sobre el tráfico que el *ISP* envíe al Sistema de Votaciones.

Reacción

La reacción principal a un ataque de esta naturaleza consiste en filtrar el tráfico del ataque permitiendo el paso sólo al tráfico de red legítimo. Esa acción que puede ser realizada por medio del firewall y de los ruteadores propuestos a incluirse en la topología de red, muestra la dificultad real de defenderse de un Ataque de Negación de Servicios: No habiendo una forma efectiva de identificar las peticiones válidas de aquellas que forman parte de un

[1] Internet Service Provider, empresa u organización que provee la salida a Internet.

ataque, se debe determinar que tráfico permitir y cual bloquear, a fin de atender peticiones legítimas y descartar las maliciosas.
Como alternativa a la compleja problemática de bloquear las peticiones de un ataque, sin afectar al tráfico valido, se han propuesto las siguientes estrategias:

Debido a que los ataques de negación de Servicios son lanzados utilizando herramientas para esos fines realizan conexiones a los servidores utilizando ciertos puertos y protocolos; de tenerse tipificado el tipo de tráfico que se genera al sistema de Votaciones, si se bloquea todo aquél tráfico que no tenga el perfil esperado, se podría mitigar el problema para dichos casos.

Otra Herramienta útil para el filtrado de tráfico ilegítimo son los Sistemas de Detección de Intrusos. Ellos por su capacidad de encontrar patrones anómalos en las comunicaciones entre máquinas, si son para red, o en el comportamiento de un Servidor si son para host; de incluirse en las subredes de los Servidores del Sistema de Votaciones, y en los servidores, podrían auxiliar a detectar aquellas peticiones que no parecieran normales. Cabe aclarar que un Sistema de detección de Intrusos no es trivial de configurarse, lleva tiempo hacerlo, pues requiere tomar muestras del comportamiento de la red y de los sistemas que protegerá, durante largos periodos de tiempo en los que estos deben estar operando normalmente. Por ello esta alternativa sería de utilidad después de que se haya tipificado perfectamente el comportamiento del Sistema de Votaciones, lo que ocurriría tras la utilización del Sistema en varias elecciones o simulacros.

En caso de que el tráfico de red que genera el ataque sea bajo un patrón muy similar al del tráfico legítimo y no se pueda diferenciar entre conexiones válidas, se puede optar por bloquear arbitrariamente un porcentaje del tráfico recibido. Esta medida garantizaría que los servidores del sistema no se dieran de baja por el agotamiento de sus recursos, sin embargo, a su vez el tráfico legitimo se vería fácilmente afectado al tener que ser rechazado junto con las peticiones maliciosas.

Seguridad para los Clientes del Sistema de Votaciones

Debido a que en las Votaciones remotas, el sistema esta abierto a recibir el voto desde cualquier máquina conectada al Internet, no se puede aplicar ningún control para la seguridad de la Infraestructura desde donde el elector emitiría su voto. Ello hace que la responsabilidad de protegerse de las amenazas inherentes a la plataforma de Internet recaigan en el Votante quien deberá tener una cultura informática avanzada a fin de no ser vulnerado fácilmente.

Algunas recomendaciones que podría auxiliar a este propósito son las siguientes:

La máquina desde donde emita su voto el votante deberá tener un sistema operativo con todas las actualizaciones de seguridad; y a su vez tendría que tener instalado un firewall personal así como software antivirus y antispyware.

El Votante debería tener instalado previamente en su navegador la llave pública de *SSL* de los servidores a fin de verificar que cuando realicen las conexiones a estos durante las votaciones, se estén contactando a los servidores reales. Esto implicaría que al tiempo que el votante recibe su identificador de votante y contraseña, recibe también en medios magnéticos las llaves públicas de los servidores del Sistema de Votaciones.

Consideraciones sobre las Votaciones Remotas

A pesar de los controles que se puedan aplicar sobre la plataforma de Votaciones, los riesgos que implican realizar una votación de esta naturaleza siguen siendo muy altos; Los controles aplicados a la Infraestructura de Servidores si bien brindan cierta seguridad no garantizan que un ataque de Negación de Servicios Distribuidos no logré su cometido de dar de baja las votaciones, sobre todo si éste se realiza con la suficiente intensidad (una gran cantidad de máquinas participan) y su sofisticación es suficiente como para no permitir distinguir entre el tráfico ilegal y el valido.

A su vez, la seguridad que se le puede ofrecer a la Infraestructura del Votante es nula, por lo que el elector esta expuesto a todas las amenazas de Internet dependiendo sólo de los recursos con los que él mismo cuente para protegerse.

Por lo anterior mientras no existan cambios fundamentales en la plataforma de Internet o en el área de la seguridad en cómputo que permita hacer de la primera una plataforma con una seguridad aceptable, pensar en implementar un Sistema de Votaciones Remotas, sería una acción imprudente en el que la postura de este trabajo es de completo rechazo.

4.3.2 Seguridad para Votaciones en Kioscos

Seguridad para los Servidores del Sistema de Votaciones

La seguridad para los servidores en este tipo de votaciones se basa también en tener un esquema de red seguro, con una topología semejante a la propuesta en las votaciones remotas, pero con una diferencia fundamental: no todo el tráfico de Internet será permitido, únicamente se aceptará aquél que provenga de ciertos lugares autorizados para realizar las votaciones. Por la poca efectividad de realizar autenticación por IP, para detectar el tráfico proveniente de los sitios autorizados, sería necesario implementar *VPNs*[2], que garanticen que sólo el tráfico de ciertos sitios sea aceptado.

Defensa contra Ataques de Negación de Servicios Distribuidos

Se aplican los mismos principios mencionados en el escenario de Votaciones Remotas; pero con las siguientes diferencias:

[2] Virtual Private Network, Canal Privado de Comunicaciones implementado por medio de criptografía sobre una red pública.

I Prevención

Dado que la infraestructura que participará en las elecciones está bien delimitada, es posible y recomendable robustecer la seguridad de ésta, a fin de que sea más difícil a un atacante comprometer el número de máquinas suficiente para lanzar un ataque de esta naturaleza.

II Reacción

Al tener una infraestructura de red de menor dimensión, es posible monitorear el tráfico de la red a fin de detectar de qué zonas de la red está llegando el tráfico malicioso para bloquearlo.

Las herramientas que permiten conectarse a los agentes instalados en las máquinas que lanzan el ataque de negación de servicios necesitan abrir algún puerto para comunicarse, si las redes participantes están lo suficientemente estructuradas, es decir, se sabe que tipo de protocolos y en que puertos se realiza la comunicación de cada zona, sería posible detectar las conexiones a dichos puertos y bloquearlas a fin de que el atacante pierda el control sobre las máquinas que comprometió.

Seguridad para los Clientes del Sistema de Votaciones

Al ser limitada la infraestructura desde la que los electores votarán, los administradores de dicha red están en posibilidad de aplicar los siguientes controles:

- Las máquinas que se utilizarán como clientes para las votaciones, se encuentran libres de Virus, cuentan con firewall personal y software antivirus y antispyware.
- La estructura de la red está protegidos por un ruteador, un firewall y un ruteador de la misma forma en la que se protege a los servidores; por lo que no se permite que ningún tipo de tráfico pueda contactar a los Clientes a no ser aquél que provenga de la *VPN* implementada para el Sistema de Votaciones. Ver Diagrama 6

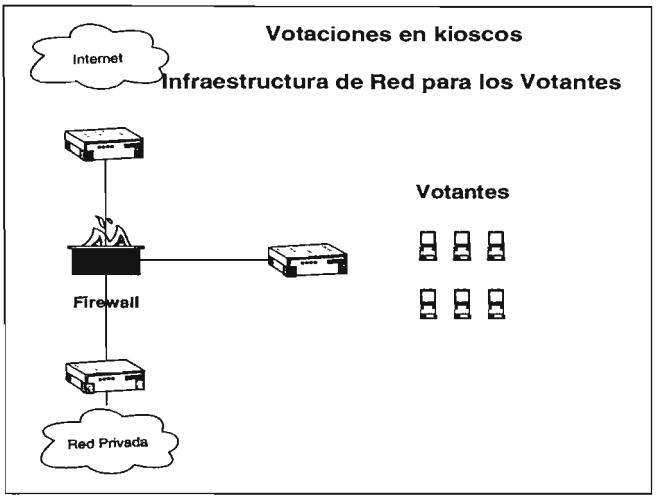


Diagrama 6

- Con las limitaciones y dificultad que implica el implementar un Sistema de Detector de Intrusos por el tiempo que lleva su configuración, es recomendable que exista uno a nivel de Red en la Infraestructura que use el elector, a fin de detectar incidentes durante la realización de las Votaciones.

Consideraciones sobre las Votaciones en Kioscos

Técnicamente de seguirse las recomendaciones planteadas es posible dotar al Sistema de Votaciones con la seguridad suficiente que vuelva factible implementar el sistema de Votaciones. Sin embargo, robustecer la seguridad de toda la infraestructura de red y computadoras involucradas, implica un esfuerzo logístico entre los administradores de las diferentes redes que puede llegar a ser muy costoso.

Un problema a considerar, es que según el esquema de votaciones en kioscos explicado en el capítulo 1, se delega la responsabilidad de los clientes del Sistema de Votaciones a diferentes instancias de cuya honestidad dependerá que realmente se protejan esos equipos; sin embargo, de no existir mecanismos de supervisión y auditoria sobre ellos, nada impediría que estos aprovecharan su control sobre los equipos que se usan como clientes para alterar los votos.

Otro aspecto a tomar en cuenta es que un esquema de esta naturaleza requeriría la utilización de *VPNs* las cuales incrementarían el costo de implementar el Sistema de Votaciones considerablemente.

De contarse con los recursos suficientes para superar las limitantes logísticas y económicas, este es un esquema que podría ser utilizado. Sin embargo, queda para analizarse su conveniencia ya que pueden existir Sistemas de Votaciones que no sólo sean menos costosos, sino que además permitan el Voto de un perfil más amplio de Votantes y no sólo de aquellos con cultura Informática.

4.3.3 Seguridad para Votaciones en Módulos

Seguridad para los Servidores del Sistema de Votaciones

Los mecanismos de seguridad para este esquema son similares, la diferencia que brindará mayor seguridad, es que el acceso al Sistema de Votaciones se realiza desde una red Interna aislada de Internet por lo que se tendría una topología de red bajo el siguiente esquema:

- La red donde se implemente el sistema estaría detrás de un ruteador principal equipo que sería el enlace con Internet y quien realizaría filtrado de paquetes; bloqueando todas las peticiones dirigidas al sistema de Votaciones.
- En una segunda capa se situaría un firewall configurado a nivel de Aplicación para el análisis de los paquetes. Este firewall que distribuiría aquellos paquetes entre la *DMZ*[3] bloquearía toda la comunicación dirigida del exterior al Sistema de Votaciones.
- En una tercera capa, se situaría antes de cada subred un ruteador que proporcionará defensa a profundidad sobre el tráfico que sea transmitido ilegalmente al Sistema de votaciones.
- Como se mencionó en el capítulo 4, se conserva la configuración de mantener un router, un firewall a nivel aplicación, y otro router entre las subredes del servidor que aloja a la entidad Verificador y la del Contador.
- Sería recomendable estructurar bien la red donde se implemente el sistema de tal forma que cada router y firewall de la red Interna, rechace el tráfico de red dirigido al Sistema de Votaciones si este no proviene de donde debería venir. De esta forma también se proporcionaría defensa contra la Red Interna.
- Como se puede observar en el Diagrama 7, el esquema es muy similar a aquél planteado para los servidores en las votaciones remotas; sólo que todo se lleva dentro de una Red Interna.

[3] *Demilitarized Zone, segmento de una red Interna accesible desde Internet.*

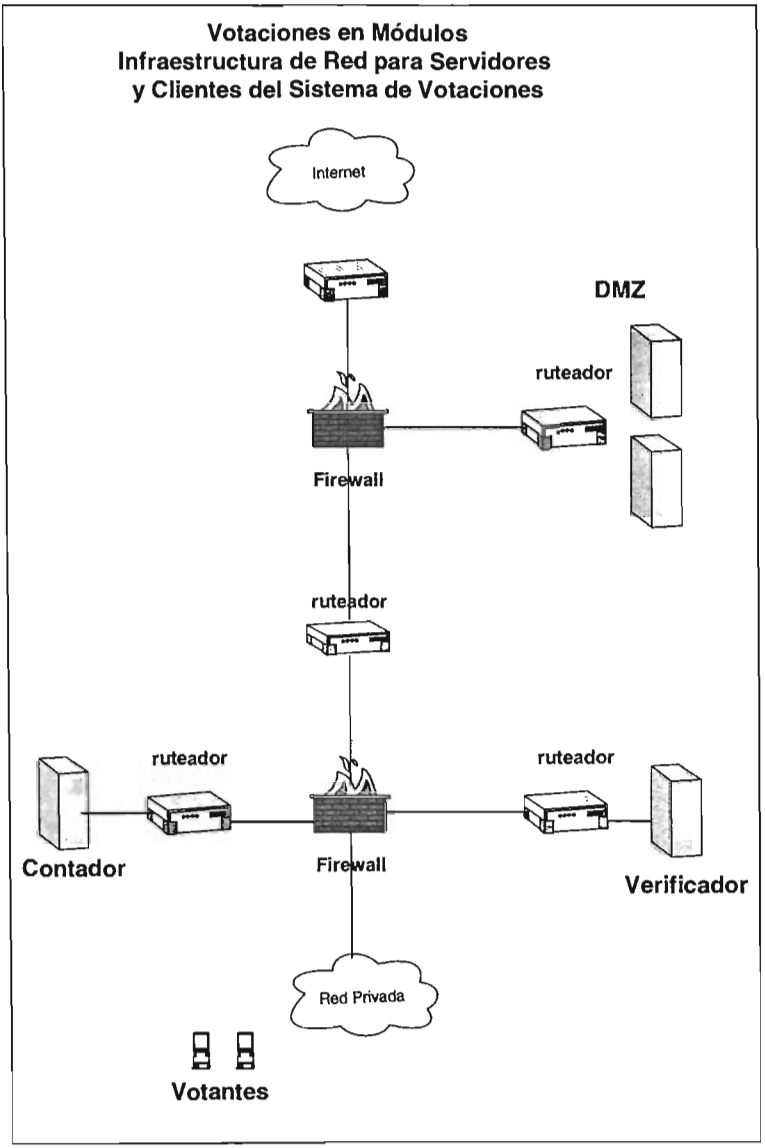

Votaciones en Módulos
Infraestructura de Red para Servidores
y Clientes del Sistema de Votaciones
Internet
DMZ
ruteador
Firewall
ruteador
ruteador
ruteador
Contador
Firewall
Verificador
Red Privada
Votantes

Diagrama 7

Defensa contra Ataques de Negación de Servicios Distribuidos

Se aplican las mismas medidas que las explicadas en el escenario de Votaciones en kioscos, la ventaja es que al tener una red más pequeña, aplicar todas las recomendaciones planteadas es más viable.

Seguridad para los Clientes del Sistema de Votaciones

Los principios para proteger los equipos que se utilizarán como clientes de votaciones, son los mismos que aquellos de las Votaciones en Kioscos, con la ventaja de que al ser más pequeña la red involucrada, existe mayor supervisión y control a fin de que a los equipos realmente se les proteja.

Consideraciones sobre las Votaciones en Módulos

Definitivamente este esquema es el más factible de implementarse, teniendo una infraestructura de menor dimensión y que además es propia, no sólo se vuelve factible que se apliquen las medidas necesarias para proteger el Sistema de Votaciones contra las amenazas existentes, además existe la posibilidad de supervisar y comprobar que se esté realizando así.

Capítulo 5 Conclusiones

Desafortunadamente para todo el paradigma de los protocolos de votaciones revisados, un problema insalvable en ellos y en el utilizado, es que se basan en autenticación por conocimiento; ante este paradigma siempre será posible la compra o coerción del secreto. Ésta situación será más fácil de llevarse a cabo si se realizan votaciones remotas, mientras que si el voto se realiza en ambientes controlados se podrían poner en marcha mecanismos para evitar la compra y coerción del voto si a la autenticación por password del votante se le añade algún control para asegurarse que el Votante es quien emitirá su voto de forma libre.

En general el uso de Internet para realizar votaciones remotas expone las elecciones a amenazas a nivel global, lo cual es un escenario alarmante. Actualmente no existen acciones que garanticen seguridad en contra de todas las posibles amenazas que implicaría poner en marcha el Sistema que se propone.

Por otro lado como se menciono de implantarse el sistema en un escenario de Votaciones en Módulos o en Kioscos, se tendría mayor control sobre los equipos de comunicaciones y sobre los clientes; esto mitigaría en buena medida las posibles amenazas que pudieran ocurrir sobre las entidades que participan en el sistema. Sin embargo, un problema que se observa en estos esquemas es que aún cuando se proteja la infraestructura para aislarla de las amenazas de Internet, ésta aún formará parte de ella y por más controles que se establezcan, la posibilidad de una infiltración siempre será latente.

Al no vislumbrarse cambios en la infraestructura de Internet en el mediano o largo plazo, los Sistemas de Votaciones que usan su plataforma, seguirán restringidos a realizarse en ambientes controlados. Sin embargo de utilizarse éste sistema, aún cuando sea en ambientes donde se pierde buena parte de la accesibilidad que en un principio era su principal ventaja y motivación. Seguirán encontrándose algunas ventajas sobre otros dispositivos de voto electrónico como lo son los *DREs*[4]:

- El Hardware de los Servidores es más confiable, las compañías que los venden llevan años diseñándolos bajo estándares de calidad que a través del tiempo han hecho innovar tecnologías que brindan alta disponibilidad.

- Es más fácil realizar auditorias sobre el Software de éste Sistema de Votaciones en Internet que utiliza software libre, que sobre los circuitos de un *DRE* y su software, el cual, generalmente es propiedad intelectual.

[4] Direct Recording Electronic, ver sección 1.2.5

- Al reaprovechar la Infraestructura y tecnología existente, cualquier Institución con una red privada podría implementar éste Sistema sin tener que Invertir en múltiples máquinas para realizar votaciones

Algunas de las Desventajas que tiene este sistema ante los *DREs* son las siguientes:

- Existen *DREs* que utilizan pilas y son portátiles por lo que no requieren de UPS[5] ni una estructura de red para funcionar.

- Las Interfaces de algunos *DREs* son más intuitivas y permiten el voto de invidentes mientras que no todos los usuarios saben utilizar Internet.

Se puede generalizar que un sistema de este tipo al reaprovechar la infraestructura de redes de computadoras, es más adecuado para entornos en los que ya exista una infraestructura de este tipo como son Empresas, Institutos, Universidades. Pero podría verse en desventaja ante otros sistemas si no existen esas condiciones.

Por último cabe recordar que la tecnología de Votaciones por Internet prácticamente no ha sido probada, por lo que antes de realizar votaciones reales con este y con cualquier sistema, es necesario que se superen una serie de pruebas que simulen las condiciones reales en las que operaría el sistema. Una estrategia muy difundida cuando se prueba una nueva tecnología de votaciones, es utilizar el sistema simultáneamente con el sistema tradicional, a fin de comparar los resultados.

Futuras Mejoras al Sistema

En el sistema presentado, no se provee alta disponibilidad la cual sería deseable; por lo que en una segunda etapa de este proyecto sería conveniente utilizar una granja de servidores con dos o más equipos para cada entidad. Ello no generaría grandes cambios en el sistema debido que tanto el Servidor Web Tomcat como el manejador de Bases de Datos PostgreSQL soportan configuraciones de esa naturaleza

A su vez sería deseable separar la base de datos del Servidor Web a fin de que se ejecuten en servidores independientes, esto disminuiría los privilegios del Administrador de dicha máquina y generaría un nuevo rol, que mitigaría los ataques por parte de las personas que tienen control sobre el Sistema al dividir los privilegios.

[5] Unbreakable Power System, Equipos que proveen de Alimentación Eléctrica Ininterrumpida.

APÉNDICE A Código Fuente del Sistema

Servlet Validador

Archivo: Validador.java

Este servlet verifica en la Base de Datos electores que las credenciales proporcionadas por el votante sean válidas, en caso de que así sea redirige al elector a una liga para descargar el applet. Además crea una sesión en el servidor web tomcat a fin de que el applet tenga los privilegios suficientes para emitir un voto.

```
package com.elliptic;
import java.io.*;
import java.util.*;
import javax.servlet.*;
import javax.servlet.http.*;
import java.sql.*;
import org.apache.log4j.Logger;
import org.apache.log4j.PropertyConfigurator;
import javax.servlet.ServletContext;
import java.math.BigInteger;
import javax.naming.*;

public class Validador extends HttpServlet implements HttpSessionListener{
public static Logger logger = Logger.getLogger(Validador.class);

 public void doGet(HttpServletRequest request, HttpServletResponse response)
  throws IOException, ServletException
  {
    try {
     Context env = (Context)new InitialContext().lookup("java:comp/env");

     /*PropertyConfigurator.configure("C:/Archivos de programa/"+
                 "Apache Software Foundation/Tomcat 5.5/"+
                 "webapps/votaciones/WEB-INF/classes/com/"+
                "elliptic/logVal.conf");*/

     String logs = (String) env.lookup("logVal");
     PropertyConfigurator.configure(logs);

     response.setContentType("text/html");
     PrintWriter out = response.getWriter();
     String ipaddress = request.getRemoteAddr();
     out.println("Acceso Invalido " + ipaddress);

     logger.info("Peticion via GET - " + ipaddress);
    }
    catch(Exception e)
  {
   logger.error("No fue posible leer parametros de contexto" + e);
  }

  }

  public void doPost(HttpServletRequest request, HttpServletResponse res)
  throws IOException, ServletException
  {
     try {

     Context env = (Context) new InitialContext().lookup("java:comp/env");

     ServletContext listado;
     listado = getServletContext();
     /*PropertyConfigurator.configure("C:/Archivos de programa/"+
                 "Apache Software Foundation/Tomcat 5.5/"+
```

```
            "webapps/votaciones/WEB-INF/classes/com/"+
            "elliptic/logVal.conf");*/


String logs = (String)env.lookup("logVal");
PropertyConfigurator.configure(logs);


String id_q, password_q, nombre_q, voto_firmac_q;
PrintWriter out = res.getWriter ();
String ipaddress = request.getRemoteAddr();
String id = request.getParameter("id_votante");
String password = request.getParameter("password");
int largo = request.getContentLength();
String px = (String)env.lookup("PKx");
String py = (String)env.lookup("PKy");
String urlPrincipal = (String)env.lookup("urlPrincipal");

if(largo>60)
{
 out.println("<html>Error las credenciales proporcionadas son incorrectas");
 out.println("<br> <a href="+urlPrincipal+"> regresar </a></html>");
 logger.warn("Envio de Credenciales exceden longitud valida - "+ipaddress);
}
else if ((id == null) || (id.trim().equals(""))||(password == null)||(password==""))
{
  out.println("<html>Error las credenciales proporcionadas son incorrectas");
  out.println("<br> <a href="+urlPrincipal+"> regresar </a></html>");
  logger.warn("Envio de Credenciales nulas - "+ipaddress);
}
else if (!((chkintegridad(id))&&(chkintegridad(password))))
{
  out.println("<html>Error las credenciales proporcionadas son incorrectas");
  out.println("<br> <a href="+urlPrincipal+"> regresar </a></html>");
  logger.warn("Envio de caracteres prohibidos - "+ipaddress);
}
else
{
 try {
 Class.forName("org.postgresql.Driver");

 }

 catch(ClassNotFoundException cnfe)
 {
   logger.error("Error controlador postgres  - " + ipaddress);
 }

 try
 {
  String url = "jdbc:postgresql://localhost:5432/electores";
  Properties props = new Properties();
  props.setProperty("user", "autenticador");
  String passautenticador = (String) env.lookup("passautenticador");
  props.setProperty("password",passautenticador);
  Connection conn = DriverManager.getConnection(url, props);

  Statement st = conn.createStatement();
  ResultSet rs = st.executeQuery("SELECT id,password," +
          "voto_firmac FROM elecciones.votantes WHERE id= " + id );

  if (rs.next())
  {
   id_q=rs.getString(1);
   password_q=rs.getString(2);
   voto_firmac_q=rs.getString(3);
   rs = st.executeQuery("SELECT chkpass_out(\'" + password + "\')");
   rs.next();
   if (password_q.equals(rs.getString(1)))
   {
```

```
if (voto_firmac_q==null)
{
   rs.close();
   st.close();
   conn.close();

   if(listado.getAttribute(id)!=null)
   {
    logger.warn("Doble registro para - "+id+" - "+ ipaddress);
    out.println("<html>Error las credenciales proporcionadas son incorrectas");
    out.println("<br> <a href="+urlPrincipal+"> regresar </a></html>");
   }
   else
   {
    listado.setAttribute(id, "registrado");
    ECDomainParameters dp = ECDomainParameters.NIST_B_283();


    //px = "992534465331641994894148591205464712070599455947319144452240157365017032373986892621 4";
    //py = "18665215641562330402440840146287408113739345097700607806465346187487228684366842990 56";

    ECPrivKey KBlindc = new ECPrivKey(dp);
    ECPubKey RBlindc = new ECPubKey(KBlindc);
    // Generacion punto R aleatorio
    while(RBlindc.W.x.val.compareTo(BigInteger.valueOf(0))==0)
    {
     KBlindc = new ECPrivKey(dp);
     RBlindc = new ECPubKey(KBlindc);
    } //ciclo valida que el componente x de RBlindc no sea cero.



    HttpSession session = request.getSession(true);
    session.setMaxInactiveInterval(300);
    session.setAttribute("KBlindc", KBlindc);
    session.setAttribute("RBlindc", RBlindc);
    session.setAttribute("elector", id);
    session.setAttribute("password",password);

    String urlContador =(String) env.lookup("urlContador");
    //Respondiendo con el Applet y sus parámetros de inicialización
    out.println("<html>");
    out.println("<head>");
    out.println("<title>Boleta de Voto</title>");
    out.println("</head><body> ");
    out.println("<center>");
    out.println("<applet code=com.elliptic.Voto.class archive=sVoto.jar width=\"100%\" height=\"100%\">");

    //out.println("<param name=\"sessionId\" value="+ session.getId()+">");
    out.println("<param name=\"sessionId\" value=" + id + ">");
    //Es enviado en formato texto el x,y del punto R Circunflejo
    out.println("<param name=\"RBlindcx\" value=" +
            RBlindc.W.x.val.toString() + ">");
    out.println("<param name=\"RBlindcy\" value=" +
            RBlindc.W.y.val.toString() + ">");
    out.println("<param name=\"PKx\" value=" + px + ">");
    out.println("<param name=\"PKy\" value=" + py + ">");
    out.println("<param name=\"urlContador\" value=" + urlContador + ">");
    out.println("</applet></center></body></html>");
    logger.info("Votante autenticado - "+id);
   }
}
else
{
 out.println(
    "<html>Error las credenciales proporcionadas son incorrectas");
 out.println("<br> <a href="+urlPrincipal+"> regresar </a></html>");
 logger.warn("Votante - "+" "+id+" - intento ilicito de acceso -"+ipaddress );
 rs.close();
```

```
            st.close();
           }

          }
          else
           {
            out.println("<html>Error las credenciales proporcionadas son incorrectas");
            out.println("<br> <a href="+urlPrincipal+"> regresar </a></html>");
            logger.info("Password erroneo para Votante - "+id+" - "+ipaddress);
            rs.close();
          st.close();
           }
          }
          else
          {
          out.println("<html>Error las credenciales proporcionadas son incorrectas");
          out.println("<br> <a href="+urlPrincipal+"> regresar </a></html>");
          logger.info("Identificador de Votante desconocido - "+ id+ " - "+ ipaddress);
          rs.close();
          st.close();
          }
        }
        catch(Exception e)
        {
          logger.error("Error " + e);
        }
      }
     }
     catch(Exception e)
      {
        logger.error("No fue posible leer parametros de contexto" + e);
      }

  }

public static boolean chkintegridad(String input)
{
  char c;
  for(int i=0; i<input.length(); i++)
{
   c = input.charAt(i);
  switch (c)
  {
    case '\':
      return false;
    case '"':
      return false;
    case ';':
      return false;
    case '\\':
      return false;
    case '':
      return false;
    }
}
    return true;

  }
   public void sessionCreated(HttpSessionEvent e)
   {}
   public void sessionDestroyed(HttpSessionEvent e) {
     HttpSession session = e.getSession(); // get handle to session.
     String id = (String)session.getAttribute("elector");
     ServletContext listado;
     listado =getServletContext();
     listado.removeAttribute(id);
    }

}
```

Servlet Firmador

Archivo: Firmador.java

Este servlet verifica que las solicitudes de voto provengan de un elector a quien se le creó una sesión en el servidor web para votar, en caso de que así sea genera la firma digital ciega del voto enmacarado que recibe del votante.

```
package com.elliptic;
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;
import java.math.BigInteger;
import com.elliptic.ECDomainParameters;
import com.elliptic.ECPrivKey;
import java.util.Properties;
import java.sql.DriverManager;
import java.sql.Statement;
import java.sql.ResultSet;
import java.sql.Connection;
import java.security.spec.ECFieldF2m;
import org.apache.log4j.PropertyConfigurator;
import org.apache.log4j.Logger;
import javax.naming.Context;
import javax.naming.InitialContext;


public class Firmador extends HttpServlet {
public static Logger logger = Logger.getLogger(Firmador.class);

 public void doPost(
  HttpServletRequest request,
  HttpServletResponse res)
  throws ServletException, IOException {
  try
  {
    HttpSession session = request.getSession(false);
    String ipaddress = request.getRemoteAddr();
   if (session != null)
   {

     /* PropertyConfigurator.configure("C:/Archivos de programa/"+
       "Apache Software Foundation/Tomcat 5.5/"+
       "webapps/votaciones/WEB-INF/classes/com/"+
      "elliptic/logFir.conf");  */

      Context env = (Context) new InitialContext().lookup("java:comp/env");
      String logs = (String)env.lookup("logFir");
      PropertyConfigurator.configure(logs);
      String key, sVoto, elector, BlindVotoh,curve,version,password;
      String BlindDsignature;
      key = (String)env.lookup("PRK");
      String versionloc = "1.00";
      String curva = "NIST_B_283";
      int i=0;
      short largo_componentes = 75;
      short largo_version =4;
      short largo_curve = 10;
      short largo_BlindVotoh= 89;

      String padding10 ="0000000000";
      BigInteger llave, Voto;
      res.setContentType("application/x-java-serialized-object");
      //se obtiene el voto protegido del applet
```

```
InputStream in = request.getInputStream();
ObjectInputStream inputFromApplet = new ObjectInputStream(in);
BlindVotoh = (String) inputFromApplet.readObject();

if (BlindVotoh.length() == largo_BlindVotoh )
{

 version = BlindVotoh.substring(i,largo_version);
 i = i + largo_version;
 curve = BlindVotoh.substring(i,i+largo_curve);
 i = i + largo_curve;
 sVoto = BlindVotoh.substring(i);

 if ((version.equals(versionloc))&&(curve.equals(curva)))
 {
   ECDomainParameters dp = ECDomainParameters.NIST_B_283();
   //Thread.sleep(10000);
   ECPrivKey KBlindc = (ECPrivKey) session.getAttribute("KBlindc");
   ECPubKey RBlindc = (ECPubKey) session.getAttribute("RBlindc");
   elector = (String) session.getAttribute("elector");
   password = (String) session.getAttribute("password");
   Voto = new BigInteger(sVoto,16);
   llave = new BigInteger(key);
   ECPrivKey sk = new ECPrivKey(dp, llave);
   BECDSA VCF = new BECDSA();
   VCF.initSignature(sk,KBlindc,RBlindc,Voto.toByteArray());
   VCF.sign();

   String url = "jdbc:postgresql://localhost:5432/electores";
   Properties props = new Properties();
   props.setProperty("user", "firmante");
   String passfirmante = (String)env.lookup("passfirmante");
   props.setProperty("password", passfirmante);
   Connection conn = DriverManager.getConnection(url, props);
   Statement st = conn.createStatement();

   ResultSet rs = st.executeQuery("SELECT voto_firmac " +
       "FROM elecciones.votantes WHERE id= " + elector );


   if((VCF.c.compareTo(BigInteger.valueOf(0))==0)||
     (VCF.d.compareTo(BigInteger.valueOf(0))==0))
   {
    logger.error("Error al generar la firma - " + ipaddress);
    OutputStream outstr = res.getOutputStream();
    ObjectOutputStream oos = new ObjectOutputStream(outstr);
    oos.writeObject(version+" "+curve);
    oos.flush();
    oos.close();

   }
   else if (rs.next())
   {
     String voto_firmac_q=rs.getString(1);
     if (voto_firmac_q==null)
     {

        logger.info("Voto Autorizado - " +elector);
        int result = st.executeUpdate("UPDATE elecciones.votantes "+
        "SET voto_firmac = \'"+VCF.c.toString(16)+" "+VCF.d.toString(16)+
        "\' WHERE id = "+elector);

        result = st.executeUpdate("UPDATE elecciones.votantes "+
        "SET password_plano = \'"+password+
        "\' WHERE id = "+elector);
        rs.close();
        st.close();
        conn.close();
        BlindDsignature=version+curve+
```

```
            padding10.substring(0,largo_componentes-VCF.c.toString(16).length())+
            VCF.c.toString(16)+
            padding10.substring(0,largo_componentes-VCF.d.toString(16).length())+
            VCF.d.toString(16);


            OutputStream outstr = res.getOutputStream();
            ObjectOutputStream oos = new ObjectOutputStream(outstr);
            oos.writeObject(BlindDsignature);
            oos.flush();
            oos.close();
          }
          else
            logger.fatal("Intentó emitir un voto adicional - "+ elector +" - " + ipaddress);
        }
        else
          logger.fatal("Sesion ilegal para votante - "+elector +" - " + ipaddress);
        }
        else
        {

         logger.fatal("Voto mal formado - " + ipaddress);
         OutputStream outstr = res.getOutputStream();
         ObjectOutputStream oos = new ObjectOutputStream(outstr);
         oos.writeObject(version+" "+curve);
         oos.flush();
         oos.close();

        }
      }
      else
      {
       logger.fatal("Voto con longitud exccesiva - " + ipaddress);
       OutputStream outstr = res.getOutputStream();
       ObjectOutputStream oos = new ObjectOutputStream(outstr);
       oos.writeObject("voto invalido b");
       oos.flush();
       oos.close();
      }
      session.invalidate();
    }
    else
    {
      logger.warn("Sesion no iniciada - " + ipaddress);
      OutputStream outstr = res.getOutputStream();
      ObjectOutputStream oos = new ObjectOutputStream(outstr);
      oos.writeObject("Sesion no Iniciada... que tratas de hacer");
      oos.flush();
      oos.close();
    }
  }
    catch (Exception e)
    {
       e.printStackTrace();
       OutputStream outstr = res.getOutputStream();
       ObjectOutputStream oos = new ObjectOutputStream(outstr);
       oos.writeObject("error firmador" + e.toString());
       oos.flush();
       oos.close();
       logger.error("error servlet Firmador" + e.toString());

    }
}


}
```

Servlet Contador

Archivo: Contador.java

El servlet contador verifica que los votos que recibe sean válidos; esto lo realiza descartando los duplicados y verificando la firma digital que se envía con estos. En caso de ser válidos, escribe los votos en la Base de Datos Votaciones.

```
package com.elliptic;
import java.io.*;
import java.util.*;
import javax.servlet.*;
import javax.servlet.http.*;
import java.sql.*;
import com.elliptic.internal.Utils;
import java.security.MessageDigest;
import java.math.BigInteger;
import org.apache.log4j.PropertyConfigurator;
import org.apache.log4j.Logger;
import javax.naming.Context;
import javax.naming.InitialContext;

public class Contador extends HttpServlet {
 public static Logger logger = Logger.getLogger(Contador.class);
 public Contador() {
  try {
   jbInit();
  }
  catch (Exception ex) {
   ex.printStackTrace();
  }
 }

 public Boolean invalido;

 public void doGet(HttpServletRequest request, HttpServletResponse response)
  throws IOException, ServletException
  {
    /*PropertyConfigurator.configure("C:/Archivos de programa/"+
              "Apache Software Foundation/Tomcat 5.5/"+
              "webapps/votaciones/WEB-INF/classes/com/"+
                 "elliptic/logCon.conf");*/
    try
    {
     Context env = (Context)new InitialContext().lookup("java:comp/env");
     String logs = (String) env.lookup("logCon");
     PropertyConfigurator.configure(logs);

     response.setContentType("text/html");
     PrintWriter out = response.getWriter();
     String ipaddress = request.getRemoteAddr();
     out.println("Acceso Invalido " + ipaddress);
     logger.info("Peticion via GET - " + ipaddress);
    }
    catch (Exception e)
    {

     logger.error(e);

    }


  }

 public void doPost(HttpServletRequest request, HttpServletResponse res)
     throws ServletException, IOException
 {
```

```
/*PropertyConfigurator.configure("C:/Archivos de programa/"+
            "Apache Software Foundation/Tomcat 5.5/"+
            "webapps/conteo/WEB-INF/classes/com/"+
            "elliptic/logCon.conf");*/


try
{
Context env = (Context) new InitialContext().lookup("java:comp/env");
String logs = (String)env.lookup("logCon");
PropertyConfigurator.configure("LogCon");
MessageDigest sha;
BigInteger AL1B,AL2B,FCB,FDB;
String ipaddress = request.getRemoteAddr();
short largo_componentes = 75;
short largo_aleatorios = 35;
short largo_candidato = 50;
short largo_version =4;
short largo_curve = 10;
short largo_Votosigned = 284;
String pxs = (String)env.lookup("PKx");
String pys = (String)env.lookup("PKy");
int i=0;
String version,curve,M,Votosigned;
byte [] M0;
invalido=invalido.FALSE;
//SE OBTIENEN LOS DATOS
String al1,al2,candidato,fc,fd;
try
{
 InputStream in = request.getInputStream();
 ObjectInputStream inputFromApplet = new ObjectInputStream(in);
 Votosigned = (String) inputFromApplet.readObject();
 if (! (Votosigned.length() == largo_Votosigned)) {
  invalido = invalido.TRUE;
  logger.fatal("Longitud de Voto invalida " + ipaddress);
 }
 else {
  version = Votosigned.substring(i, largo_version);
  i = i + largo_version;
  curve = Votosigned.substring(i, i + largo_curve);
  i = i + largo_curve;
  candidato = Votosigned.substring(i, i + largo_candidato);
  i = i + largo_candidato;
  al1 = Votosigned.substring(i, i + largo_aleatorios);
  i = i + largo_aleatorios;
  al2 = Votosigned.substring(i, i + largo_aleatorios);
  i = i + largo_aleatorios;
  fc = Votosigned.substring(i, i + largo_componentes);
  i = i + largo_componentes;
  fd = Votosigned.substring(i, i + largo_componentes);
  i = i + largo_componentes;
  if (! ( (chkintegridad(al1)) && (chkintegridad(al2))
       && (chkintegridad(candidato)) && (chkintegridad(fc))
       && (chkintegridad(fd))))
  {
   invalido = invalido.TRUE;
   logger.fatal("Caracteres invalidos en Votosigned - "+ipaddress);
  }
  else if ( (! (invalido.booleanValue())) && (version.equals("1.00")) &&
     (curve.equals("NIST_B_283")))
    {
    ECDomainParameters dp = ECDomainParameters.NIST_B_283();
    BigInteger px, py;
    ECPubKey pk;
    ECPointF2m pkpoint;

    px = new BigInteger(pxs);
```

```
py = new BigInteger(pys);
F2m auxx = new F2m(px);
F2m auxy = new F2m(py);
pkpoint = new ECPointF2m(auxx, auxy);
pk = new ECPubKey(dp, pkpoint);

AL1B = new BigInteger(al1, 16);
AL2B = new BigInteger(al2, 16);
M = candidato.trim();
candidato = M;
FCB = new BigInteger(fc, 16);
FDB = new BigInteger(fd, 16);

try {
 sha = MessageDigest.getInstance("SHA-256");
 M = M.concat(AL1B.toString());
 sha.update(M.getBytes());
 M0 = sha.digest();

 M = Utils.intArrayToString(Utils.toIntArray(M0));
 M = M.concat(AL2B.toString());

 ECDSA Mf = new ECDSA(FCB, FDB);
 Mf.initVerify(pk);
 Mf.update(M.getBytes());
 if (Mf.verify()) {
  Class.forName("org.postgresql.Driver");
  String url = "jdbc:postgresql://localhost:5432/votaciones";
  Properties props = new Properties();
  props.setProperty("user", "escrutador");
  String passescrutador= (String)env.lookup("passescrutador");
  props.setProperty("password",passescrutador);
  Connection conn = DriverManager.getConnection(url, props);
  Statement st = conn.createStatement();
  boolean inserta =
     st.execute("INSERT INTO elecciones.votos" +
           " (al1,al2,candidato,firma) VALUES " +
           "(\"" + AL1B.toString(16) + "\"," + "\"" +
           AL2B.toString(16) + "\"," + "\"" + candidato +
           "\"," + "\"" + FCB.toString(16) +
           "--" + FDB.toString(16) + "\")");

  st.close();
  conn.close();

  OutputStream outstr = res.getOutputStream();
  ObjectOutputStream oos = new ObjectOutputStream(outstr);
  oos.writeObject("La firma es valida");
  oos.flush();
  oos.close();
  logger.info("Voto Almacenado");
 }

 else {
  OutputStream outstr = res.getOutputStream();
  ObjectOutputStream oos = new ObjectOutputStream(outstr);

  oos.writeObject("firma erronea");
  oos.flush();
  oos.close();
  logger.fatal("Voto Ilegal - " + ipaddress);
 }
}

catch (Exception e) {
 invalido = invalido.TRUE;

 OutputStream outstr = res.getOutputStream();
 ObjectOutputStream oos = new ObjectOutputStream(outstr);
 oos.writeObject("Error");
```

```
        oos.flush();
        oos.close();
        String passescrutador= (String)env.lookup("passescrutador");
        logger.error(" Error en Servlet Contador - " +passescrutador+ e);
       }

     }
     else {
      OutputStream outstr = res.getOutputStream();
      ObjectOutputStream oos = new ObjectOutputStream(outstr);
      oos.writeObject("Parametros recibidos no validos");
      oos.flush();
      oos.close();
     }
    }
   }
      catch (Exception e)
{


 logger.error(e);
 e.printStackTrace();
 OutputStream outstr = res.getOutputStream();
 ObjectOutputStream oos = new ObjectOutputStream(outstr);
 oos.writeObject("error firmador " + e.toString());
 oos.flush();
 oos.close();

 }
   }
   catch (Exception e)
    {

     logger.error(e);
     e.printStackTrace();
     OutputStream outstr = res.getOutputStream();
     ObjectOutputStream oos = new ObjectOutputStream(outstr);
     oos.writeObject("error firmador " + e.toString());
     oos.flush();
     oos.close();

  }
}

 public static boolean chkintegridad(String input)
 {
  char c;
  for(int i=0; i<input.length(); i++)
 {
   c = input.charAt(i);
  switch (c)
  {
   case '\'':
     return false;
   case '"':
     return false;
   case ';':
     return false;
   case '\\':
     return false;
   }
 }
   return true;
 }

 private void jbInit() throws Exception {
 }
}
```

Applet Voto

Archivo: Voto.java

El Applet Voto proporciona una interfaz gráfica para ejercer el voto, para ello se encarga de obtener la firma digital del firmador y de enviar ésta junto con el voto al Contador. Esto implica ocuparse del enmascaramiento del voto para obtener una firma digital ciega y del desenmascarado de la firma digital ciega para obtener una firma digital normal que pueda ser enviada junto con el voto.

```
/*
 * Boleta de Voto
  Clase General Voto
  contiene a la Clase VotoDialogo encargada de realizar la interfaz grafica
  y a la clase firmaciega encargada de generar la firma ciega

 */
package com.elliptic;
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;
import java.math.BigInteger;
import java.security.SecureRandom;
import java.security.MessageDigest;
import java.lang.Boolean;
import com.elliptic.F2m;
import com.elliptic.ECDomainParameters;
import com.elliptic.ECPointF2m;
import com.elliptic.internal.Utils;
import java.net.*;
import java.io.*;

public class Voto extends JApplet
{
  public Boolean AccesoValido;
  public String FirmaC, Firma;
  public void votar(String M) {

    //llave publica se recibe al momento de descargar el applet
    //^R tambien se recibe al momento de descargar el applet
    ECDomainParameters dp = ECDomainParameters.NIST_B_283();
    BigInteger puntox, puntoy, fc, fd;
    ECPubKey RBlindc, pk;
    ECPointF2m aux, RBlind;
    BigInteger AL1, AL2;
    BigInteger alfa;
    BigInteger beta;
    BigInteger M1, XRinv;
    byte[] M0;
    String candidato, BlindVotoh, BlindDsignature, Votosigned;
    String padding50, padding10;
    String version, curve;
    String curva="NIST_B_283";
    String versionloc="1.00";
    candidato = M;
    int i=0;
    short largo_componentes = 75;
    short largo_aleatorios = 35;
    short largo_version =4;
    short largo_curve = 10;
    short largo_BlindDsignature= 164;
    padding10 = "0000000000";
    //padding de 10 ceros
    padding50 = "                                                  ";
    //padding de 50 espacios
```

```
String Sauxx = getParameter("RBlindcx");
String Sauxy = getParameter("RBlindcy");
String urlContador = getParameter("urlContador");
puntox = new BigInteger(Sauxx);
puntoy = new BigInteger(Sauxy);
F2m auxx = new F2m(puntox);
F2m auxy = new F2m(puntoy);

aux = new ECPointF2m(auxx, auxy);
RBlindc = new ECPubKey(dp, aux);

Sauxx = getParameter("PKx");
Sauxy = getParameter("PKy");
puntox = new BigInteger(Sauxx);
puntoy = new BigInteger(Sauxy);
auxx.val = puntox;
auxy.val = puntoy;

aux = new ECPointF2m(auxx, auxy);
pk = new ECPubKey(dp, aux);
//   W.x= wx.toByteArray();


//ECPubKey pk = new ECPubKey(dp,W);

MessageDigest sha;
// Aqui se recibe el RBlindp para generar el RBlind
try {
SecureRandom rnd = SecureRandom.getInstance("SHA1PRNG");
alfa = new BigInteger(dp.m, rnd);
AL1 = new BigInteger(128, rnd);
AL2 = new BigInteger(128, rnd);

alfa = alfa.mod(dp.r);


  alfa = alfa = new BigInteger(dp.m,rnd);
RBlind = (ECPointF2m) RBlindc.dp.E.mul(alfa, RBlindc.W);
//Aqui ya se computó Rblind
//Aqui ya se computó Rblind
XRinv = Utils.OS2IP(Utils.FE2OSP(RBlind.x));
XRinv = XRinv.modInverse(dp.r);
beta = Utils.OS2IP(Utils.FE2OSP(RBlindc.W.x)).mod(dp.r);
beta = beta.multiply(XRinv).mod(dp.r);
beta = beta.multiply(alfa).mod(dp.r);
//aqui Beta ya fue calculado como la parte X de RBlindc por
//el inverso de la parte X de RBlind por alfa
try {
  sha = MessageDigest.getInstance("SHA-256");
  M = M.concat(AL1.toString());
  sha.update(M.getBytes());
  M0 = sha.digest();
  //Hash de la Concatenacion cadena del Candidato con numero aleatorio

  M = Utils.intArrayToString(Utils.toIntArray(M0));
  M = M.concat(AL2.toString());
  sha.update(M.getBytes());
  M0 = sha.digest();
  //Concatenacion del Hash anterior con otra cadena aleatoria

  M1 = Utils.OS2IP(Utils.revIntArray(Utils.toIntArray(M0)));

  M1 = M1.multiply(beta).mod(dp.r);
  M1 = M1.multiply(alfa.modInverse(dp.r)).mod(dp.r);

  BlindVotoh = versionloc + curva + padding10.substring(0,largo_componentes -
     M1.toString(16).length()) + M1.toString(16);
```

```
URL url = new URL(getCodeBase(), "Firmador");
URLConnection con = url.openConnection();

con.setDoOutput(true);
con.setDoInput(true);
con.setUseCaches(false);
con.setRequestProperty(
   "Content-Type",
   "application/x-java-serialized-object");

//envia peticion al Firmador


OutputStream outstream = con.getOutputStream();
ObjectOutputStream oos = new ObjectOutputStream(outstream);
oos.writeObject(BlindVotoh);
oos.flush();
oos.close();

// recibe respuesta
InputStream instr = con.getInputStream();
ObjectInputStream inputFromServlet = new ObjectInputStream(instr);
BlindDsignature = (String) inputFromServlet.readObject();
inputFromServlet.close();
instr.close();

version = BlindDsignature.substring(i, largo_version);
i = i +largo_version;
curve = BlindDsignature.substring(i, i+largo_curve);
i = i + largo_curve;
FirmaC = BlindDsignature.substring(i);


if ( (BlindDsignature.length() == largo_BlindDsignature) && (version.equals(versionloc) ) &&
   (curve.equals(curva)))
{

  fc = new BigInteger(FirmaC.substring(0, largo_componentes),16);
  fd = new BigInteger(FirmaC.substring(largo_componentes,
                          largo_componentes*2),16);

  BECDSA FdC = new BECDSA(fc, fd);
  FdC.initVerify(pk, M1.toByteArray());
  if (FdC.verify())
   // Verifica la firma con el enmascarado
  {
   System.out.println("firma ciega valida ");
    AccesoValido = AccesoValido.TRUE;
  }
  else {
   System.out.println("error en firma del verificador");
   AccesoValido = AccesoValido.FALSE;
  }
  FdC.c = Utils.OS2IP(Utils.FE2OSP(RBlind.x)).mod(dp.r);
  FdC.d = FdC.d.multiply(beta.modInverse(dp.r));
  FdC.d = FdC.d.mod(dp.r);

  ECDSA FirmaVoto = new ECDSA(FdC.c, FdC.d);
  FirmaVoto.initVerify(pk);
  FirmaVoto.update(M.getBytes());
  if (FirmaVoto.verify()&&(AccesoValido.booleanValue()))
  {
   System.out.println("firma digital valida ");

   FirmaC= fc.toString(16)+" "+fd.toString(16);
   Firma = FirmaVoto.c.toString(16) + " " + FirmaVoto.d.toString(16);
   AccesoValido = AccesoValido.TRUE;

   //CODIGO PARA ENVIAR EL VOTO
```

```
        URL url2 = new URL(urlContador);

        URLConnection con2 = url2.openConnection();
        con2.setDoOutput(true);
        con2.setDoInput(true);
        con2.setUseCaches(false);

        //envia el voto para ser registrado

        Votosigned = version + curve + padding50.substring(candidato.length()) +
          candidato +
          padding10.substring(0,largo_aleatorios - AL1.toString(16).length()) +
          AL1.toString(16) +
          padding10.substring(0,largo_aleatorios - AL2.toString(16).length()) +
          AL2.toString(16) +
          padding10.substring(0,largo_componentes -
                    FirmaVoto.c.toString(16).length()) +
          FirmaVoto.c.toString(16) +
          padding10.substring(0,largo_componentes -
                    FirmaVoto.d.toString(16).length()) +
          FirmaVoto.d.toString(16);



        con2.setRequestProperty(
        "Content-Type",
        "application/x-java-serialized-object");

//envia al contador


          OutputStream outstream2 = con2.getOutputStream();
          ObjectOutputStream oos2 = new ObjectOutputStream(outstream2);
          oos2.writeObject(Votosigned);
          oos2.flush();
          oos2.close();

          InputStream instrc = con2.getInputStream();
          ObjectInputStream inputFromServletc = new ObjectInputStream(instrc);
          String resultc = (String) inputFromServletc.readObject();
          inputFromServletc.close();
          instrc.close();

          System.out.println(resultc);



        }
        else {
          System.out.println("Firma Digital apocrifa");
          AccesoValido = AccesoValido.FALSE;
        }
      }

      else {
        System.out.println("Firma Digital mal Formada");
        AccesoValido = AccesoValido.FALSE;
      }
    }
    catch (Exception e)
    {
      System.out.println("el error " + e.toString());
    }

  } catch (Exception e)
    {
      System.out.println("el error " + e.toString());
    }

  }
```

```
public class DialogoVoto
      extends JPanel {
    JLabel label;
    JInternalFrame frame;
    String simpleDialogDesc = "Candidatos Elegibles";

    public DialogoVoto(JInternalFrame frame) {

     super(new BorderLayout());
     this.frame = frame;
     JLabel title;
      //Create the components.
     JPanel choicePanel = createSimpleDialogBox();


     title = new JLabel("Dar click en el boton \"Votar\" "
                 + " una vez que se haya elegido el candidato.",
                 JLabel.CENTER);

     label = new JLabel("Sistema de Votaciones en Internet", JLabel.CENTER);
     label.setBorder(BorderFactory.createEmptyBorder(10, 10, 10, 10));
     choicePanel.setBorder(BorderFactory.createEmptyBorder(20, 20, 5, 20));

     //Lay out the main panel.
     add(title, BorderLayout.NORTH);
     add(label, BorderLayout.SOUTH);
     add(choicePanel, BorderLayout.CENTER);
    }

    void setLabel(String newText) {
     label.setText(newText);
    }

    private JPanel createSimpleDialogBox() {
     final int numButtons = 7; //numero de opciones

     JRadioButton[] radioButtons = new JRadioButton[numButtons];

     final ButtonGroup group = new ButtonGroup();

     JButton voteButton = null;

     radioButtons[0] = new JRadioButton(
        "<html><font color=blue>Candidato A</font></html>");
     radioButtons[0].setActionCommand("Candidato A");

     radioButtons[1] = new JRadioButton(
        "<html><font color=blue>Candidato B</font></html>");
     radioButtons[1].setActionCommand("Candidato B");

     radioButtons[2] = new JRadioButton(
        "<html><font color=blue>Candidato C</font></html>");
     radioButtons[2].setActionCommand("Candidato C");

     radioButtons[3] = new JRadioButton(
        "<html><font color=blue>Candidato D</font></html>");
     radioButtons[3].setActionCommand("Candidato D");

     radioButtons[4] = new JRadioButton(
        "<html><font color=blue>Candidato E</font></html>");
     radioButtons[4].setActionCommand("Candidato E");

     radioButtons[5] = new JRadioButton(
        "<html><font color=blue>Candidato F</font></html>");
     radioButtons[5].setActionCommand("Candidato F");

     radioButtons[6] = new JRadioButton(
        "<html><font color=blue>Candidato G</font></html>");
     radioButtons[6].setActionCommand("Candidato G");
```

```
//Aqui se Añaden las opciones


    for (int i = 0; i < numButtons; i++) {
     group.add(radioButtons[i]);
    }

    //Select the first button by default.
    radioButtons[0].setSelected(true);

    voteButton = new JButton("Votar");

    voteButton.addActionListener(new ActionListener() {
     public void actionPerformed(ActionEvent e) {
      String candidato = group.getSelection().getActionCommand();

      //ok dialog
      int n = JOptionPane.showConfirmDialog(frame,
                                "<html><font color=black>" +
                                "Esta seguro que desea ejercer su voto por: \n" +
                                candidato + "\n ",
                                "Confirmar Voto"
                                , JOptionPane.YES_NO_OPTION);
      if (n == JOptionPane.YES_OPTION) {
       votar(candidato);
       if(AccesoValido.booleanValue())
       {
        setLabel("Su Voto ha Sido ejercido.");
        JOptionPane.showMessageDialog(frame, "Su Voto ha sido ejercido");
       }
       else
       {
        setLabel("Error, su Voto no se ejerció");
        JOptionPane.showMessageDialog(frame, "Error, su Voto no se ejerció");
       }
       frame.dispose();
       stop();
      }
      else if (n == JOptionPane.NO_OPTION) {
       setLabel("Elegir Votante");
      }

      return;
     }
    });

    return createPane(simpleDialogDesc + ":", radioButtons, voteButton);
   }

   private JPanel createPane(String description, JRadioButton[] radioButtons,
                             JButton showButton) {
    int numChoices = radioButtons.length;
    JPanel box = new JPanel();
    JLabel label = new JLabel(description);

    box.setLayout(new BoxLayout(box, BoxLayout.PAGE_AXIS));
    box.add(label);

    for (int i = 0; i < numChoices; i++) {
     box.add(radioButtons[i]);
    }

    JPanel pane = new JPanel(new BorderLayout());
    pane.add(box, BorderLayout.NORTH);
    pane.add(showButton, BorderLayout.SOUTH);

    return pane;

   }
```

```
    }

  public void creaGUI() {

   int x, y;

   JFrame.setDefaultLookAndFeelDecorated(true);
   JDialog.setDefaultLookAndFeelDecorated(true);

   JInternalFrame frame = new JInternalFrame("Boleta de Voto");

     JDesktopPane padre = new JDesktopPane();

   getContentPane().add(padre, java.awt.BorderLayout.CENTER);

   padre.add(frame);

   frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

   Container contentPane = frame.getContentPane();
   contentPane.setLayout(new GridLayout(1, 1));
   contentPane.add(new DialogoVoto(frame));


   try {
    frame.pack();
    frame.setMaximum(true);
    frame.setVisible(true);
   }
   catch (Exception e) {
    System.err.println("No se pudo graficar la interfaz");
   }


   };

   public void init() {

   }

   public void start() {

    try {

     javax.swing.SwingUtilities.invokeAndWait(new Runnable() {
      public void run() {
       try {
        String sessionId = getParameter("sessionId");
        if (! (sessionId == null)) {
         AccesoValido = AccesoValido.TRUE;
         creaGUI();
        }
        else {
         AccesoValido = AccesoValido.FALSE;
         setLayout(new GridBagLayout());
         Label mensaje = new Label("Sesion no iniciada", Label.CENTER);
         mensaje.setFont(new Font("SansSerif", Font.BOLD, 14));
         add(mensaje);

        }

       }
       catch (Exception e) {
        System.err.println("Error de comunicacion con el Servidor: " +
                           e);
       }
      }
     });
    }
```

```
    catch (Exception e) {
      System.err.println("No se pudo iniciar la interfaz: " + e);
    }
  }

  public void stop() {

    if (AccesoValido.booleanValue()) {

      Label Label1 = new Label();
      Label1.setText("Firma Digital Ciega: ");
      Label Label2 = new Label(FirmaC.substring(0,FirmaC.indexOf(" ")));
      Label Label3 = new Label(FirmaC.substring(FirmaC.indexOf(" ")+1));
      Label Label4 = new Label("Firma Digital del Voto: ");
      Label Label5 = new Label(Firma.substring(0,Firma.indexOf(" ")));
      Label Label6 = new Label(Firma.substring(Firma.indexOf(" ")+1));
      Label Label7 = new Label(
        "IMPRIMA ESTA PAGINA; ES EL COMPROBANTE DE SU VOTO");

      setLayout(new GridLayout(15, 1));
      add(Label1);
      add(Label2);
      add(Label3);
      add(Label4);
      add(Label5);
      add(Label6);
      add(Label7);

    }
    else {
      setLayout(new GridBagLayout());
      Label mensaje = new Label("Sesion no iniciada", Label.CENTER);
      mensaje.setFont(new Font("SansSerif", Font.BOLD, 14));
      add(mensaje);
    }

  }
}
```

Clase BECDSA

Archivo: BECDSA.java

Esta clase fue creada con base en el Código de la Clase ECDSA perteneciente al paquete JBorzoi. Permite la generación y verificación de firmas digitales ciegas.

```
/*
Blind ECDSA
Modificación a la Clase ECDSA del Paquete JBorzoi
Para la realización de Firmas Digitales Ciegas
 */

public class BECDSA {
  public BigInteger c;
  public BigInteger d;

  private ECPrivKey s;
  private ECPrivKey KB;
  private ECPubKey W;
  private ECPubKey RB;
  private int f[];
  private byte f2[];
  private MessageDigest sha;

  private BigInteger[] ECSP_DSA() {
    BigInteger sig[] = {
       BigInteger.valueOf(0), BigInteger.valueOf(0)};
    ECPrivKey u;
    ECPubKey V;

     /* Genera otro par de llaves u,v para el calculo del punto aleatorio R
        que auxilia en la generación de la firma. Lo calcula a partir de los
        mismos parametros de la curva */

     u = KB;
     V = RB;
     sig[0] = Utils.OS2IP(Utils.FE2OSP(V.W.x)).mod(s.dp.r);
     //V.W.x es el componente X del punto aleatorio R
     BigInteger uinv = u.s.modInverse(s.dp.r);
     //uinv es el k para generar el R
     BigInteger temp =
        Utils.OS2IP(f).add(s.s.multiply(sig[0]).mod(s.dp.r)).mod(s.dp.r);
     // esto es la multoplicacion de cs + e
     sig[1] = (uinv.multiply(temp)).mod(s.dp.r);
     // esto lo multiplica kinv( cs + e )


    return sig;
  }

  private boolean ECVP_DSA() {
    if (! ( ( (BigInteger.valueOf(1).compareTo(c) <= 0)
          & (c.compareTo(W.dp.r) < 0))
        & ( (BigInteger.valueOf(1).compareTo(d) <= 0)
          & (d.compareTo(W.dp.r) < 0)))) {
     //System.out.println("c" + c.toString());
     //System.out.println("d" + d.toString());
     //System.out.println("r" + W.dp.r.toString());
     //System.out.println("no 1");
     return false;
    }

    BigInteger h = d.modInverse(W.dp.r);
    BigInteger h1 = Utils.OS2IP(f).multiply(h).mod(W.dp.r);
```

```
  BigInteger h2 = c.multiply(h).mod(W.dp.r);

  ECPoint P = W.dp.E.add(W.dp.E.mul(h1, W.dp.G), W.dp.E.mul(h2, W.W));

  if (P.isZero()) {
    //System.out.println("no 2");
    return false;
  }

  BigInteger i = Utils.OS2IP(Utils.FE2OSP(P.x)).mod(W.dp.r);

  if (c.compareTo(i) == 0) {
    //System.out.println("P" + i.toString());
    //System.out.println("C" + c.toString());
    return true;

  }
  else {
    System.out.println("no 3");
    System.out.println("P " + i.toString());
    System.out.println("C " + c.toString());
    return false;
  }
}

public BECDSA() {
  try {
    jbInit();
  }
  catch (Exception ex) {
    ex.printStackTrace();
  }
}

public BECDSA(BigInteger c, BigInteger d) {
  this.c = c;
  this.d = d;
}

public void initSignature(ECPrivKey s, ECPrivKey KBlindp,
                          ECPubKey RBlindp, byte[] ftemp) throws
    NoSuchAlgorithmException {
  //sha = MessageDigest.getInstance("SHA-1");
  this.s = (ECPrivKey) s.clone();
  this.KB = (ECPrivKey) KBlindp.clone();
  this.RB = (ECPubKey) RBlindp.clone();
  this.f2 = ftemp;
}

public void initVerify(ECPubKey W, byte[] ftemp) throws
    NoSuchAlgorithmException {
  //sha = MessageDigest.getInstance("SHA-1");
  this.W = (ECPubKey) W.clone();
  this.f2 = ftemp;
}

// public void update(byte[] data) {
//   sha.update(data);
// }

public void sign() {
//    f = Utils.toIntArray(f2);
  f = Utils.revIntArray(Utils.toIntArray(f2));
  BigInteger[] sig = ECSP_DSA();
  c = sig[0];
  d = sig[1];
}

public boolean verify() {
//    f = Utils.toIntArray(f2);
```

```
    f = Utils.revIntArray(Utils.toIntArray(f2));
  return ECVP_DSA();
 }



 public String toString() {
  String str = new String("c:").concat(c.toString(16)).concat("\n");
  str = str.concat("d:").concat(d.toString(16)).concat("\n");

  return str;
 }

 protected Object clone() {
  return new BECDSA();
 }

 private void jbInit() throws Exception {
 }

}
```

Clase sessionlis

Archivo: sessionlis.java

Esta clase es utilizada para destruir sesiones en el servidor web Tomcat. Esto resulta necesario y se realiza cada vez que un usuario ha emitido su voto y cuando éste ha excedido el tiempo definido para ejercerlo.

```
/*

Clase para Destruir sesiones de usuarios que ya hayan votado
o que hayan excedido el tiempo para hacerlo

*/

package com.elliptic;
import javax.servlet.http.HttpSession;
import javax.servlet.http.HttpSessionListener;
import javax.servlet.http.HttpSessionEvent;
import javax.servlet.ServletContext;
import javax.servlet.*;

public class sessionlis implements HttpSessionListener

 {
  ServletContext listado;


  public void sessionCreated(HttpSessionEvent e)
  {}
  public void sessionDestroyed(HttpSessionEvent e) {
   HttpSession session = e.getSession();
   String id = (String)session.getAttribute("elector");
   listado = session.getServletContext();
   listado.removeAttribute(id);
  }
}
```

Página Principal del Sistema

Archivo: index.html

El código html básicamente es una forma en la que se captura el identifcador del votante y el password para redirigir estos parámetros al Servlet Autenticador.

```
<html>
<head>
<title>Sistema de Votaciones </title>
<body bgcolor="FFFFFF">
<center><table BORDER=0 CELLSPACING=0 CELLPADDING=0 WIDTH="680" BGCOLOR="#FFFFFF" >
<tr>
<br>
<br>
<br>
<td COLSPAN="3">
<table BORDER=0 CELLSPACING=0 CELLPADDING=4 WIDTH="100%" >
<tr>
<td VALIGN=BOTTOM>
<table BORDER=0 CELLSPACING=0 CELLPADDING=0 WIDTH="100%" >
<tr>
<td ALIGN=LEFT WIDTH="1%"></td>
<td ALIGN=RIGHT VALIGN=BOTTOM NOWRAP></td>
</tr>
</table>
</td>
</tr>
<tr>
<td BGCOLOR="#A0B8C8">
<div align="center"><b><font face="Arial"><font color="#0000FF"><font size=+1>Bienvenido 
al Sistema de Votaciones</font></font></font></b></div>
</td>
</tr>
</table>
</td>
</tr>

<tr>
<td COLSPAN="3">
<table BORDER=0 CELLSPACING=2 CELLPADDING=3 WIDTH="100%" >
<tr>
<td ALIGN=CENTER><b><font face="arial"><font color="#008080">Debe registrarse
para poder acceder al sitio.</font></font></b></td>
</tr>
</table>
</td>
</tr>

<tr>
<td VALIGN=TOP COLSPAN="3">
<table BORDER=0 CELLSPACING=2 CELLPADDING=0 WIDTH="98%" >
<tr>
  <td VALIGN=TOP WIDTH="6%">
  <table BORDER=0 CELLSPACING=0 CELLPADDING=2 WIDTH="100%" >
  <tr>
   <td ALIGN=CENTER><b><nobr><font face="Arial"></font></nobr></b></td>
  </tr>
   <tr>
   <td ALIGN=CENTER><b></b></td>
  </tr>
  </table>
<!-- property_promo -->
```

```
        <table BORDER=0 CELLSPACING=0 CELLPADDING=0 >
          <tr>
            <td ALIGN=RIGHT VALIGN=TOP> </td>
            <td ALIGN=LEFT VALIGN=TOP> </td>
          </tr>
        </table>
<!-- property_promo --></td>

        <td WIDTH="1%"> </td>

        <td ALIGN=LEFT VALIGN=TOP width="93%">
          <form method=post action="https://Verificador:8443/votaciones/Validador">
<table BORDER=0 CELLSPACING=0 CELLPADDING=2 WIDTH="100%" BGCOLOR="#A0B8C8" >
<tr>
<td>
<table BORDER=0 CELLSPACING=0 CELLPADDING=2 WIDTH="100%" BGCOLOR="#EEEEEE" >
<tr>
<td ALIGN=CENTER BGCOLOR="#FFFFFF">
<table BORDER=0 CELLSPACING=6 CELLPADDING=6 WIDTH="100%" BGCOLOR="#FFFFFF" >
<tr BGCOLOR=#EEEEEE >
<td ALIGN=CENTER>


<br><nobr><font size=2 face="arial"><font color=#000000> 
Proporcione sus credenciales para ejercer su voto <br><br>
 </font></font></nobr>

<table BORDER=0 CELLSPACING=0 CELLPADDING=4 >

<table BORDER=0 CELLSPACING=0 CELLPADDING=2 >
<tr>
  <td ALIGN=RIGHT NOWRAP><font face="arial"><font size=-1>Identificador de Elector </font></font></td>

<td><input name="id_votante" type=text size="16" MAXLENGTH="20"></td>
</tr>

<tr>
<td ALIGN=RIGHT NOWRAP><font face="arial"><font size=-1>Contraseña:</font></font></td>

<td><input name="password" type=password size="16" maxlength="32"></td>
</tr>

<tr>
  <td ALIGN=CENTER COLSPAN="2" NOWRAP><font face="arial"></font></td>
</tr>

<tr>
<td> </td>

<td>
<INPUT type="submit" value="Enviar">
<INPUT type="reset" value="Borrar">
                              </td>
</tr>
</table>
<b><font face="arial"><font color="#000000">
 </font></b>
<br><nobr><font face="arial"><font color=#000000><font size=-1> 
  El acceso a este Sistema esta restringido a electores validos. <br>
 </font></font></nobr>
</td>
</td>
</tr>

<tr>
    <td ALIGN=CENTER NOWRAP BGCOLOR="#EEEEEE"><font face="arial"></font></td>
</tr>
</table>
</td>
</tr>
```

```
<tr BGCOLOR="#EEEEEE">
  <td ALIGN=CENTER VALIGN=TOP><font face="arial"></font></td>
</tr>
</table>
</td>
</tr>
</table>
</td>
</tr>
</table>
</form>
 </td>
</tr>
<tr>
  <td COLSPAN="3">
 <table WIDTH="100%" >
  <tr>
  <td ALIGN=CENTER height="2"> </td>
  </tr>
  </table>
</td>
</tr>
</table>
</td>
</tr>

<tr>
<td COLSPAN="3"></td>
</tr>
</table></center>

<center>
<p>
</center>

<center><table BORDER=0 CELLSPACING=0 CELLPADDING=0 >
<tr>
    <td ALIGN=CENTER VALIGN=BOTTOM WIDTH="100%"><font face="arial"><font color="#8D8D8D"></font></font></td>
</tr>
</table></center>

</body>
</html>
```

Archivo de configuración del Sitio Virtual del Verificador

Archivo: web.xml

En este archivo de configuración se declaran los servlets, el tiempo en que expira una sesión y las llaves y passwords que utiliza el servidor web del Verificador

```
<?xml version="1.0" encoding="ISO-8859-1"?>

<!DOCTYPE web-app
   PUBLIC "-//Sun Microsystems, Inc.//DTD Web Application 2.3//EN"
   "http://java.sun.com/dtd/web-app_2_3.dtd">

<web-app>


   <!-- General description of your web application -->

   <display-name>My Web Application</display-name>
   <description>
    This is version X.X of an application to perform
    a wild and wonderful task, based on servlets and
    JSP pages.  It was written by Dave Developer
    (dave@mycompany.com), who should be contacted for
    more information.
   </description>


   <!-- Context initialization parameters that define shared
       String constants used within your application, which
       can be customized by the system administrator who is
       installing your application.  The values actually
       assigned to these parameters can be retrieved in a
       servlet or JSP page by calling:

           String value =
             getServletContext().getInitParameter("name");

       where "name" matches the <param-name> element of
       one of these initialization parameters.

       You can define any number of context initialization
       parameters, including zero.
   -->

   <context-param>
     <param-name>webmaster</param-name>
     <param-value>otoniel4@prodigy.net.mx</param-value>
     <description>
       The EMAIL address of the administrator to whom questions
       and comments about this application should be addressed.
     </description>
   </context-param>


   <!-- Servlet definitions for the servlets that make up
       your web application, including initialization
       parameters.  With Tomcat, you can also send requests
       to servlets not listed here with a request like this:

         http://localhost:8080/{context-path}/servlet/{classname}

       but this usage is not guaranteed to be portable.  It also
       makes relative references to images and other resources
```

```
    required by your servlet more complicated, so defining
    all of your servlets (and defining a mapping to them with
    a servlet-mapping element) is recommended.

    Servlet initialization parameters can be retrieved in a
    servlet or JSP page by calling:

      String value =
       getServletConfig().getInitParameter("name");

    where "name" matches the <param-name> element of
    one of these initialization parameters.

    You can define any number of servlets, including zero.
-->
        <listener>
                <listener-class>
                com.elliptic.sessionlis
                </listener-class>
        </listener>
 <servlet>
  <servlet-name>Validador</servlet-name>
  <description>
   Este servlet autentica a los electores
  </description>
  <servlet-class>com.elliptic.Validador</servlet-class>
 </servlet>

 <servlet>
  <servlet-name>Firmador</servlet-name>
  <description>
   Este servlet Genera la firma para que los electores ejerzan su voto
  </description>
  <servlet-class>com.elliptic.Firmador</servlet-class>
 </servlet>

 <servlet-mapping>
  <servlet-name>Validador</servlet-name>
  <url-pattern>/Validador</url-pattern>

 </servlet-mapping>
 <servlet-mapping>
  <servlet-name>Firmador</servlet-name>
  <url-pattern>/Firmador</url-pattern>
 </servlet-mapping>

<env-entry>
   <description>Punto X de la llave publica</description>
   <env-entry-name>PKx</env-entry-name>
   <env-entry-value>99253446533164199489414859120546471207059945594731914445224015736501703237398689262l4</env-
entry-value>
   <env-entry-type>java.lang.String</env-entry-type>
</env-entry>

<env-entry>
   <description>Punto Y de la llave publica</description>
   <env-entry-name>PKy</env-entry-name>
   <env-entry-value>18665215641562330402440840146287408113739345097700607806465346187487228684366842990561</env-
entry-value>
   <env-entry-type>java.lang.String</env-entry-type>
</env-entry>

<env-entry>
   <description>Llave Privada</description>
   <env-entry-name>PRK</env-entry-name>
   <env-entry-value>3155759748671277911711270957196094196459521276250221736352274934698759524619232113141</env-
entry-value>
   <env-entry-type>java.lang.String</env-entry-type>
</env-entry>
```

```
<env-entry>
   <description>Log Firmador</description>
   <env-entry-name>logFir</env-entry-name>
   <env-entry-value>F:/Archivos de programa/Apache Software Foundation/Tomcat 5.5/webapps/votaciones/WEB-
INF/classes/com/elliptic/logFir.conf</env-entry-value>
   <env-entry-type>java.lang.String</env-entry-type>
</env-entry>


<env-entry>
   <description>Log Validador</description>
   <env-entry-name>logVal</env-entry-name>
   <env-entry-value>F:/Archivos de programa/Apache Software Foundation/Tomcat 5.5/webapps/votaciones/WEB-
INF/classes/com/elliptic/logVal.conf</env-entry-value>
   <env-entry-type>java.lang.String</env-entry-type>
</env-entry>

<env-entry>
   <description>Pasword Firmante</description>
   <env-entry-name>passfirmante</env-entry-name>
   <env-entry-value>smvm!nsfa%05</env-entry-value>
   <env-entry-type>java.lang.String</env-entry-type>
</env-entry>

<env-entry>
   <description>Pasword Autenticador</description>
   <env-entry-name>passautenticador</env-entry-name>
   <env-entry-value>mstl#ts026</env-entry-value>
   <env-entry-type>java.lang.String</env-entry-type>
</env-entry>

<env-entry>
   <description>URL Pagina Principal</description>
   <env-entry-name>urlPrincipal</env-entry-name>
   <env-entry-value>http://verificador:8080/</env-entry-value>
   <env-entry-type>java.lang.String</env-entry-type>
</env-entry>

<env-entry>
   <description>URL Contador</description>
   <env-entry-name>urlContador</env-entry-name>
   <env-entry-value>https://Contador:8443/conteo/Contador</env-entry-value>
   <env-entry-type>java.lang.String</env-entry-type>
</env-entry>


  <!-- Define mappings that are used by the servlet container to
       translate a particular request URI (context-relative) to a
       particular servlet.  The examples below correspond to the
       servlet descriptions above.  Thus, a request URI like:

         http://localhost:8080/{contextpath}/graph

       will be mapped to the "graph" servlet, while a request like:

         http://localhost:8080/{contextpath}/saveCustomer.do

       will be mapped to the "controller" servlet.

       You may define any number of servlet mappings, including zero.
       It is also legal to define more than one mapping for the same
       servlet, if you wish to.
  -->

  <session-config>
    <session-timeout>5</session-timeout>    <!-- 30 minutes -->
  </session-config>


</web-app>
```

Archivo de configuración del Sitio Virtual del Contador

Archivo: web.xml

En este archivo de configuración se declaran los servlets, el tiempo en que expira una sesión y las llaves y passwords que utiliza el servidor web del Contador

```
<?xml version="1.0" encoding="ISO-8859-1"?>

<!DOCTYPE web-app
   PUBLIC "-//Sun Microsystems, Inc.//DTD Web Application 2.3//EN"
   "http://java.sun.com/dtd/web-app_2_3.dtd">

<web-app>


   <!-- General description of your web application -->

   <display-name>My Web Application</display-name>
   <description>
    This is version X.X of an application to perform
    a wild and wonderful task, based on servlets and
    JSP pages.  It was written by Dave Developer
    (dave@mycompany.com), who should be contacted for
    more information.
   </description>


   <!-- Context initialization parameters that define shared
        String constants used within your application, which
        can be customized by the system administrator who is
        installing your application.  The values actually
        assigned to these parameters can be retrieved in a
        servlet or JSP page by calling:

            String value =
              getServletContext().getInitParameter("name");

        where "name" matches the <param-name> element of
        one of these initialization parameters.

        You can define any number of context initialization
        parameters, including zero.
   -->

   <context-param>
     <param-name>webmaster</param-name>
     <param-value>otoniel4@prodigy.net.mx</param-value>
     <description>
       The EMAIL address of the administrator to whom questions
       and comments about this application should be addressed.
     </description>
   </context-param>


   <!-- Servlet definitions for the servlets that make up
        your web application, including initialization
        parameters.  With Tomcat, you can also send requests
        to servlets not listed here with a request like this:

          http://localhost:8080/{context-path}/servlet/{classname}

        but this usage is not guaranteed to be portable.  It also
```

```
    makes relative references to images and other resources
    required by your servlet more complicated, so defining
    all of your servlets (and defining a mapping to them with
    a servlet-mapping element) is recommended.

    Servlet initialization parameters can be retrieved in a
    servlet or JSP page by calling:

      String value =
        getServletConfig().getInitParameter("name");

    where "name" matches the <param-name> element of
    one of these initialization parameters.

    You can define any number of servlets, including zero.
  -->



  <servlet>
    <servlet-name>Contador</servlet-name>
    <description>
      Servlet para realizar conteo de los votos
    </description>
    <servlet-class>com.elliptic.Contador</servlet-class>
  </servlet>


  <servlet-mapping>
    <servlet-name>Contador</servlet-name>
    <url-pattern>/Contador</url-pattern>

  </servlet-mapping>


<env-entry>
    <description>Punto X de la llave publica</description>
    <env-entry-name>PKx</env-entry-name>
    <env-entry-value>992534465331641994894148591205464712070599455947319144452240157365017032373986892621 4</env-
entry-value>
    <env-entry-type>java.lang.String</env-entry-type>
</env-entry>

<env-entry>
    <description>Punto Y de la llave publica</description>
    <env-entry-name>PKy</env-entry-name>
    <env-entry-value>18665215641562330402440840146287408113739345097700607806465346187487228684366842990 56</env-
entry-value>
    <env-entry-type>java.lang.String</env-entry-type>
</env-entry>

<env-entry>
    <description>Log Contador</description>
    <env-entry-name>logCon</env-entry-name>
    <env-entry-value>C:/Archivos de programa/Apache Software Foundation/Tomcat 5.5/webapps/conteo/WEB-
INF/classes/com/elliptic/logCon.conf</env-entry-value>
    <env-entry-type>java.lang.String</env-entry-type>
</env-entry>

<env-entry>
    <description>Pasword Escrutador</description>
    <env-entry-name>passescrutador</env-entry-name>
    <env-entry-value>elumieqh=oa9</env-entry-value>
    <env-entry-type>java.lang.String</env-entry-type>
</env-entry>


  <!-- Define mappings that are used by the servlet container to
```

```
    translate a particular request URI (context-relative) to a
    particular servlet.  The examples below correspond to the
    servlet descriptions above.  Thus, a request URI like:

      http://localhost:8080/{contextpath}/graph

    will be mapped to the "graph" servlet, while a request like:

      http://localhost:8080/{contextpath}/saveCustomer.do

    will be mapped to the "controller" servlet.

    You may define any number of servlet mappings, including zero.
    It is also legal to define more than one mapping for the same
    servlet, if you wish to.
  -->

  <session-config>
    <session-timeout>5</session-timeout>    <!-- 30 minutes -->
  </session-config>


</web-app>
```

Archivos de Configuración de las Bitácoras de la biblioteca logger

Archivos: logCon.conf y logVer.conf (para el Contador y Verificado respectivamente)

En este archivo de configuración se declara donde se almacenarán los datos del sistema de bitácoras.

```
log4j.rootLogger=DEBUG, R
# Pattern to output the caller's file name and line number.
log4j.appender.R=org.apache.log4j.RollingFileAppender
#El valor de log4.appender.R.File indica donde y con que nombre se almacena la bitácora en el Sistema de Archivos
#Es el único valor que cambia entre los sitios web del Contador y Verificador
log4j.appender.R.File=C:/Archivos de programa/Apache Software Foundation/Tomcat 5.5/webapps/conteo/WEB-
INF/classes/com/elliptic/logs/Contador.log
log4j.appender.R.MaxFileSize=100KB
# Keep one backup file
log4j.appender.R.MaxBackupIndex=1
log4j.appender.R.layout=org.apache.log4j.PatternLayout
#log4j.appender.R.layout.ConversionPattern=%d %p %m%n
log4j.appender.R.
```

BIBLIOGRAFÍA

[1] Acupoll
http://www.accupoll.com/TheAccuPollAdvantage/Accessibility/

[2] Bautista Osorno Moisés
Tesis de Maestría, Implementación de un sistema de cifrado de llave pública basado en el problema del logaritmo discreto para curvas elípticas
UNAM 2002

[3] Brittain, Ian F. Darwin
Tomcat: The Definitive Guide
O'Reilly 2003

[4] Bruce Momjian
PostgreSQL Introduction and Concepts
Addison Wesley 2001

[5] ByWenbo Mao
Modern Cryptography: Theory and Practice
Prentice Hall July 2003

[6] Certicom
ECC tutorial
http://www.certicom.com/index.php?action=ecc,ecc_tutorial

[7] Certicom
Why ECC is the next generation of public key cryptography

[8] CIS
Electronic Voting
http://theory.lcs.mit.edu/~cis/voting/voting.html

[9] Cranor Lorrie Faith
Electronic Voting Computerized polls may save money, protect privacy
http://www.acm.org/crossroads/xrds2-4/voting.html

[10] Cranor Lorrie Faith
Sensus
http://lorrie.cranor.org/voting/sensus/

[11] Dravis Group
Open Source Software
Case Studies Examining its Use
Abril 2003
http://www.postgresql.org/files/about/casestudies/OpenSourceSoftware_Dravis.pdf

[12] Diebold
http://www.diebold.com/dieboldes/accuvote_tsx.htm

[13] Electronic Commision of India
Electronic Voting Machine
http://www.eci.gov.in/EVM/index.htm

[14] E-Vote Vendors Hand OverSoftware
http://www.wired.com/news/print/0,1294,65490,00.html

[15] Evans David and Nathanael Paul
Election Security Perception and Reality
University of Virginia
lEEE Security and Privacy 2004 Vol. 2. N. 1

[16] Fischer A. Eric
Election Rerform and Electronic Voting Systems (DREs)
Analysis of Security Issues
CRS Report For Congress
Noviembre 2003

[17] Fujioka, Okamoto, and K. Ohta "A Practical Secret Voting Scheme for Large Scale Elections," *Advances in Cyptology* - AUSCRYPT '92

[18] Ford Fessenden
Counting the vote: the machine
New Focus on punch-card system
http://www.notablesoftware.com/Press/Fessenden.html

[19] Garfinkel Simpson
Web Security, Privacy & Commerce, 2nd Edition
O'Reilly 2001

[20] Gong Li et al
Inside Java™ 2 Platform Security: Architecture, API Design, and Implementation, Second Edition
Addison Wesley 2003

[21] Hall Marty
Servlets and JavaServerPages
Prentice Hall 2002

[22] Hankerson Darrel
Guide to Elliptic Curve Cryptography
Springer
2004

[23] Hash Collision Q&A
Cryptography Research News 2004
http://www.cryptography.com/cnews/hash.html

[24] Hebrew University of Jerusalem, Israel
Advanced Topics in Computer Security
Electronic Voting Protocols And Schemes
2002

[25] Hill David L. and. Rubin Aviel D
E-Voting Security
IEEE Security and Privacy 2004 Vol. 2. N. 1

[26] Informe Final del Simposio acerca de urnas electrónicas
http://www.iedf.org.mx/DEOyGE/Simposio/Informe_final.pdf

[27] Jefferson et al
A security Analysis of the Secure Electronic and Registration and Voting Experiment (SERVE)
January 2004

[28] Jefferson David
Requirements for Electronic and Internet Voting Systems in Public Elections
August 29 2001
http://web.mit.edu/6.857/OldStuff/Fall01/handouts/L03-jefferson1.ppt

[29] Jones W. Douglas
A Brief Illustrated History of Voting
http://www.cs.uiowa.edu/~jones/voting/pictures/

[30] Joseph P. Harris, Professor and Practioner:
Government, Election Reform, and the Votomatic
http://bancroft.berkeley.edu/ROHO/Vote/

[31] Menezes et al
Handbook of Applied Cryptography
CRC Press
1996

[32] Mercuri Rebecca
Florida 2002: Sluggish Systems, Vanishing Votes
Inside Risks
http://www.notablesoftware.com/Papers/Florida2002.pdf.

[33] Mirkovic Jelena et al
Internet Denial of Service : Attack and Defense Mechanisms
Prentice Hall
2004

[34] Morrie Gasser
Building a Secure Computer System
1987

[35] NIST
An Introduction to Computer Security:
The NIST Handbook

[36] NIST
Brief Comments on Recent Cryptanalytic Attacks on SHA-1
2005
http://csrc.nist.gov/news-highlights/NIST-Brief-Comments-on-SHA1-attack.pdf

[37] NIST
Recommendation for Key
Management – Part 1: General
Elaine Barker et al
2005

[38] Oaks Scott
Java Security Second Edition
O'Reilly May 2001

[39] Pfleeger Charles
Secure in Computing third edition
Prentice Hall 2002

[40] Randall James and Szydlo Michael
RSA Laboratories
Agosto 2004
http://www.rsasecurity.com/rsalabs/node.asp?id=2738

[41] Report of the National Workshop on Internet Voting
March 2001

[42] Residual Votes Attributables to Technology
An Assessment of the Reliability of Existing Voting Equipments
Caltech/Mit Voting Technology Project
March 30 2001

[43] Ronald L. Rivest
Electronic Voting
http://theory.lcs.mit.edu/~rivest/Rivest-ElectronicVoting.pdf

[44] Rosing Michael
Implementig Elliptic Curve Cryptography
Manning 1999

[45] Rostovtsev Alexander el al.
Elliptic Curve Ordered Digital Signature
Department of Information Security of Computer Systems
St. Petersburg State Polytechnic University 2004

[46] RSA
The RSA Challenge Numbers
http://www.rsasecurity.com/rsalabs/node.asp?id=2093

[47] Rubin Aviel et al
Analysis of an Electronic Voting System
Febrary 2004

[48] Rubin Aviel
Security Considerations for Remote Electronic Voting over the Internet
http://avirubin.com/e-voting.security.pdf

[49] Stadler Markus et al.
Fair Blind Signatures
Springer-Verlag, 1995.
http://www.ubilab.org/publications/print_versions/pdf/sta95.pdf

[50] Summers, Rita C
Secure Computing, Threats and Safeguards
Mc Graw Hill

[51] Torres Wilbert; Alberto Aguirre
"La Caida del Voto Electrónico"
Enfoque n. 528, abril 2004

[52] The Independent Commission on Alternative Methods
Elections in the 21st Century: From paper ballot to e-voting
January 2002 (83-95p)

[53] Xiaoyun Wang, et al
Collisions for Hash Functions
Agosto 2004
http://eprint.iacr.org/2004/199.pdf