Automatización Web del Proceso de Votación de las Elecciones de la EPN Utilizando Esquema de Seguridad de Firma Ciega

Web Automation of EPN's Electoral Voting Process Using Blind Signature Security Scheme

ARTICLE HISTORY

Received 24 March 2022
Accepted 02 May 2022

Jose Azadobay
Departamento de Informática y Ciencias de la Computación
Escuela Politécnica Nacional
Quito, Ecuador
jose.azadobay@epn.edu.ec

Michael Morales
Departamento de Informática y Ciencias de la Computación
Escuela Politécnica Nacional
Quito, Ecuador
michael.morales@epn.edu.ec

Hernán Ordoñez
Departamento de Informática y Ciencias de la Computación
Escuela Politécnica Nacional
Quito, Ecuador
hernan.ordonez@epn.edu.ec

Carlos Montenegro
Departamento de Informática y Ciencias de la Computación
Escuela Politécnica Nacional
Quito, Ecuador
carlos.montenegro@epn.edu.ec

Automatización Web del Proceso de Votación de las Elecciones de la EPN Utilizando Esquema de Seguridad de Firma Ciega

Web Automation of EPN's Electoral Voting Process Using Blind Signature Security Scheme

Jose Azadobay
Departamento de Informática y Ciencias de la Computación
Escuela Politécnica Nacional
Quito, Ecuador
jose.azadobay@epn.edu.ec

Michael Morales
Departamento de Informática y Ciencias de la Computación
Escuela Politécnica Nacional
Quito, Ecuador
michael.morales@epn.edu.ec

Hernán Ordoñez
Departamento de Informática y Ciencias de la Computación
Escuela Politécnica Nacional
Quito, Ecuador
hernan.ordonez@epn.edu.ec

Carlos Montenegro
Departamento de Informática y Ciencias de la Computación
Escuela Politécnica Nacional
Quito, Ecuador
carlos.montenegro@epn.edu.ec

Resumen— En la actualidad, la Escuela Politécnica Nacional tiene un proceso electoral que se lo realiza de manera manual. Por lo tanto, representa un trabajo de gestión considerable a la hora de ser llevado a cabo. Además, dada la naturaleza de los procesos manuales, está sujeto a errores humanos. Como solución a estas y otras problemáticas, el presente trabajo plantea su automatización a través de un sistema web de votación electrónica. El sistema propuesto implementa un esquema de seguridad de firmado ciego, para controlar tanto la privacidad como la validez de los votos. El desarrollo se lo realizó bajo el marco de trabajo de Scrum, debido a su ajuste a equipos de desarrollo pequeños y a su enfoque en la entrega de software funcional en cortos periodos de tiempo. El sistema implementa una arquitectura Modelo-Vista-Controlador, teniendo el desarrollo de la vista en Angular; el controlador en la plataforma .NET Framework y, finalmente, el modelo en SQL Server. Por otra parte, el sistema ha sido sometido a pruebas de usabilidad y funcionalidad, con lo que se determinó que es óptimamente usable y cumple satisfactoriamente con el 100 % de los requisitos de usuario obtenidos.

Palabras Clave— Votación Electrónica, Firma Ciega, RSA, Scrum

Abstract—Currently, the Escuela Politécnica Nacional has an electoral process that is done manually. Therefore, it represents a considerable management work when it comes to being carried out. Also, given the nature of manual processes, it is subject to human errors. As a solution to these and other problems of the electoral process in force in the institution, the present work proposes its automation through an electronic voting system. The proposed system implements a blind signing security scheme to control both the privacy of the voter and the validity of the votes. The development was carried out under the Scrum framework, due to its adaptation to small development teams and its focus on delivering functional software in short periods of time. The system implements a Model-View-Controller architecture, having the development of the view in the JavaScript framework, Angular, the controller in Microsoft .NET Framework and finally the model in SQL Server. On the other hand, the system has been subjected to usability and functionality tests, so it was determined that it is optimally usable and satisfactorily meets 100% of the user requirements obtained.

Keywords— Electronic Voting, Blind Signing, RSA, Scrum

I. INTRODUCCIÓN

Con el desarrollo de las Tecnologías de la Información y las Comunicaciones (TIC), el sistema educativo nacional y global ha tenido grandes cambios, de los cuales, el tecnológico ha sido uno de los más representativos [1]. Estos cambios implican nuevas formas de preparar, adquirir y transmitir la información [2].

A pesar del crecimiento del uso de las TIC, existen procesos en los cuales no han tenido el impacto esperado, como lo es el caso de la gestión de procesos electorales [3]. El proceso electoral que actualmente rige en la mayoría de las instituciones de educación superior y específicamente en la Escuela Politécnica Nacional (EPN), se lo realiza de manera manual, por lo tanto, está expuesto a fallas de tipo humano.

El desarrollo de un sistema de votación electrónica, que permita la ejecución más eficiente del proceso electoral de la EPN, contribuirá en la disminución de los factores del error humano y los tiempos en la obtención de resultados. Además, aporta positivamente a la preservación del ambiente ya que se elimina el uso de papeletas físicas.

El objetivo principal del presente trabajo es automatizar el proceso de votación de las elecciones de la EPN a través de un sistema de votación electrónica, con el fin de mejorar sustancialmente su desempeño.

Con el fin de garantizar el anonimato y la integridad del voto en el sistema propuesto, se implementa un esquema de firmado ciego, el cual se basa en tener una entidad independiente que se encargue de certificar la validez de un voto, sin necesidad de conocer la identidad del votante. De esta manera, se tiene un voto certificado y anónimo.

El aplicativo web del sistema propuesto está desarrollado en la versión 10 del framework de JavaScript, Angular. La lógica del sistema está implementada bajo la plataforma .Net de Microsoft y bajo esta misma plataforma se maneja la comunicación entre el aplicativo web y la lógica del negocio. Además, el sistema está construido bajo el marco de trabajo de desarrollo ágil, SCRUM.

A. SITUACIÓN ACTUAL

Actualmente, los procesos electorales en la EPN se los conlleva de la manera tradicional, es decir, son netamente manuales. Todo el proceso requiere del compromiso de un contingente humano desde la organización y designación de la Junta electoral hasta el final del proceso con la proclamación de resultados [4].

De acuerdo con el último estatuto de la EPN reformado el 24 de octubre de 2019 [5], la institución está estructurada, a nivel directivo, por:

- Consejo Politécnico
- Consejo de Docencia
- Consejo de Investigación, Innovación y Vinculación
- Consejos de Facultad
- Consejos de Instituto
- Consejos de Departamento
- Consejo Directivo de la Escuela de Formación de Tecnólogos

B. ALCANCE

El presente proyecto se llevará a cabo con base en ciertos puntos tomados de la ISO/TS 54001:2019. La norma comprende 8 subprocesos que garantizan el correcto desempeño de un organismo electoral [6], estos son:

- Inscripción electoral
- Inscripción de organizaciones políticas y de candidatos
- Logística electoral
- Emisión del voto
- Escrutinio y declaración de resultados
- Educación electoral
- Fiscalización del financiamiento de campañas electorales
- Resolución de conflictos electorales

De los subprocesos mencionados anteriormente, el sistema a implementar ayuda con la automatización de la emisión del voto y, el escrutinio y declaración de resultados. Cabe mencionar que el sistema únicamente emitirá los resultados obtenidos en el proceso de escrutinio, mismos que deberán ser necesariamente analizados por el personal miembro de junta para su respectiva declaración. Además, el sistema ayudará en parte al subproceso de logística electoral.

El subproceso de emisión del voto establecido en esta ISO, menciona que el voto deberá ser secreto, por tanto, para cumplir con este punto, se ha optado por la solución tecnológica del firmado ciego, el cual será detallado más adelante en el documento.

ISSN:1390-9266 e-ISSN:1390-9134 LAJC 2022
DOI: 10.5281/zenodo.6762958

C. METODOLOGÍA

Para la parte del desarrollo de la aplicación web, se utilizará la metodología ágil de desarrollo SCRUM, por ser un marco de trabajo que permite automatizar procesos que pueden llegar a ser complicados [7]. Este es el caso, puesto que el desarrollo de un sistema de voto electrónico puede llegar a tener un nivel de complejidad elevado ya que se debe garantizar la seguridad durante todo el ciclo de vida del proyecto, debe ser robusto, transparente y eficiente. Se utilizarán varias técnicas como patrones de diseño y técnicas de seguridad de datos a nivel de aplicación.

El grupo de trabajo que llevará a cabo la realización del proyecto propuesto en este documento está conformado por cuatro personas. Por tanto, al ser un grupo pequeño, SCRUM es de gran utilidad ya que este se enfoca principalmente en el desarrollo de proyectos que conlleva un recurso humano relativamente pequeño y manejable no mayor a 8 personas [8].

Además, gracias a la experiencia compartida en la implementación de sistemas similares [9], se puede confirmar que el uso de la metodología Scrum, resulta favorable para el desarrollo del sistema propuesto en el presente documento, ya que permite un manejo apropiado de las expectativas del cliente, basado en resultados tangibles.

D. HERRAMIENTAS

El proyecto descrito en este documento utiliza una arquitectura de tres capas utilizando el patrón de diseño Modelo Vista Controlador (MVC), el cual es un patrón de diseño de software que separa los componentes de aplicación en tres niveles: Modelos o base de datos, Vista o aspecto visual y controlador o lógica. En los cuales, cada uno de ellos tiene responsabilidades específicas [10]. La arquitectura se la puede observar en la Figura 1.

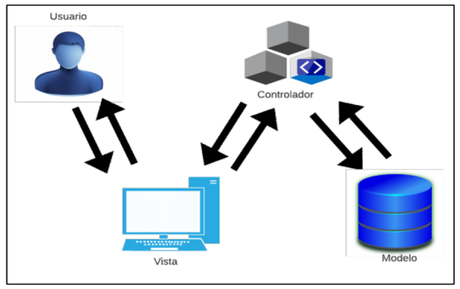

Fig. 1. Arquitectura del sistema

E. ESQUEMA DE SEGURIDAD DE FIRMA CIEGA

La firma digital, introducida inicialmente por Chaum [11], se utilizó en un principio para diseñar el primer protocolo de dinero electrónico. Posteriormente, se utilizó para verificar los votos en los esquemas de votación electrónica. Una firma digital estándar debe proporcionar ciertas características para garantizar comunicaciones digitales fiables:

- Autenticación: Todo el mundo puede verificar a la entidad u origen de la firma digital.
- Integridad: El mensaje recibido ya firmado es exactamente igual al mensaje enviado.
- No repudio: El firmante no puede negar ser el autor de un mensaje firmado válido y verificado.

En un proceso de firma digital ciega participan dos usuarios, el firmante y el cliente. El firmante firma el mensaje y lo cifra con su clave privada, por lo que él es el único que puede generarlo. El destinatario del mensaje puede verificar fácilmente la firma utilizando la clave pública del firmante. Del mismo modo, el firmante no puede negar la firma, porque si esta se puede verificar con su clave pública, significa que la firma ha sido generada con la clave privada que solo el firmante conoce [12]. Mediante el uso de firmas digitales ciegas, una institución autorizada puede firmar digitalmente una serie de datos (por ejemplo, votos) sin conocer el contenido de los datos.

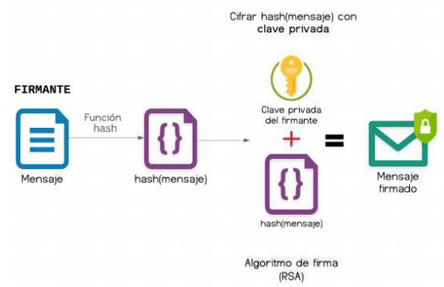

Fig. 2. Proceso de firma ciega

Por otra parte, el receptor puede verificar la firma con la clave pública del emisor o firmante de la siguiente manera:

ISSN:1390-9266 e-ISSN:1390-9134 LAJC 2022
DOI: 10.5281/zenodo.6762958

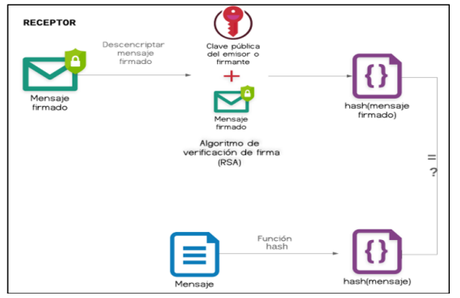

Fig. 3. Verificación de la firma

II. ANÁLISIS Y DISEÑO

A. REQUISITOS DEL SISTEMA

Para el levantamiento de los requisitos del sistema, se ha llevado a cabo entrevistas con miembros de la Federación de Estudiantes de la Escuela Politécnica Nacional (FEPON) y representantes de asociaciones de estudiantes de varias facultades. De las cuales se ha podido determinar los siguientes requerimientos:

- Ingresar al sistema con usuario y clave
- Gestionar diferentes tipos de elecciones
- Gestionar elecciones
- Gestionar escaños
- Configurar postulantes a cargos
- Configurar listas de candidatos
- Configurar procesos electorales
- Importar padrón por proceso electoral
- Enviar mails masivos a los usuarios nuevos
- Gestionar usuarios y personas
- Emitir voto
- Consultar validez del voto

B. MÓDULOS DEL SISTEMA

Con base en los requerimientos obtenidos en las entrevistas con los interesados, el sistema se dividió en cuatro módulos principales listados a continuación:

- Módulo de Configuración
- Módulo de Procesos
- Módulo de Elecciones
- Módulo de Resultados

C. DISEÑO DE INTERFAZ

Para las interfaces del módulo de configuración, se tendrá un estándar. En la parte superior, la ruta de la pantalla en donde se encuentra, esto servirá para que el usuario se ubique de mejor manera en el sistema. Seguido, se tendrá un cuadro de búsqueda, en el cual se podrán buscar registros según parámetros importantes de cada entidad.

Estas pantallas también tendrán un botón para crear nuevos registros, el cual abrirá otra pantalla auxiliar con los campos respectivos de cada entidad, el usuario deberá llenar los campos y guardar el nuevo registro. En las columnas de la tabla se tendrán botones de acción sea para eliminar, editar los registros algún otro tipo de acción implementado.

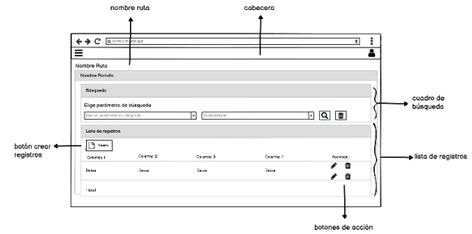

Fig. 4. Mockup de catálogos del sistema

D. ARQUITECTURA DEL VOTO PROPUESTO

La arquitectura propuesta para el proceso de emisión de voto consta de cinco fases y tres entidades participantes: el elector, un esquema de identificación (EI) y otro de votación (EV) como se puede observar en la Figura 5. Además, se considera que los canales de comunicación para el proceso son seguros. En la primera fase, una vez que el elector genera su voto (v) la aplicación, del lado del cliente, lo encripta con AES y genera una máscara (m) mediante una función hash del voto ya cifrado.

Luego, se envía el voto encriptado, la máscara generada, y el token de identificación del elector al llamado esquema de identificación. Este proceso se lo realiza a nivel del backend, en primer lugar, se verifica si el usuario está habilitado para votar, es decir, que aún no haya votado y que conste dentro del padrón electoral.

Si es así, se procede a firmar el voto cifrado + la máscara y se envía este voto firmado F(v',m) al elector. En esta parte, es donde se utiliza el concepto de firma ciega ya que la entidad que firma el voto no conoce realmente lo que está firmando pues el contenido del voto está cifrado y concatenado a una máscara.

Una vez que el elector fue autorizado para votar y ha recibido su voto firmado, lo envía al esquema de votación junto con la máscara, el voto cifrado y la llave utilizada para cifrar/descifrar el voto (keyAES). En el esquema de votación primero se verifica la validez de la firma digital utilizando la llave pública del esquema de identificación y el algoritmo de descifrado RSA explicado en apartados anteriores. Si la firma es válida, quiere decir que tanto el voto cifrado como la máscara no han sido alterados, entonces se procede a descifrar el voto con la ayuda de la llave AES que ha sido enviada desde el lado del cliente.

Una vez que tiene el voto descifrado, se almacenan las opciones que hayan sido elegidas y la máscara. La máscara sirve como identificador único de voto ya que solo es conocida por el elector y almacenada en este único caso. Cabe mencionar que de ninguna manera se puede vincular al elector con el voto o con la máscara debido a que en este instante no se tiene alguna información del elector.

Finalmente, se envía un mensaje de que el voto fue exitoso y el elector guarda su máscara para poder verificar que su voto no haya sido alterado. La verificación del voto se la podrá hacer en cualquier momento luego de que la elección haya finalizado.

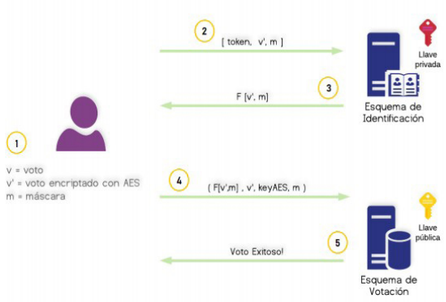

Fig. 5. Arquitectura de voto propuesto

III. IMPLEMENTACION Y PRUEBAS

A. ARQUITECTURA WEB DEL SISTEMA

Como ya se ha mencionado, la arquitectura seleccionada para este sistema está basada en MVC (modelo-vista- controlador). Para el modelo se ha utilizado como gestor de base de datos SQL Server. El controlador se desarrolló utilizando el framework .Net y el lenguaje de programación C#. Para el desarrollo de las vistas y aplicativo web, que se muestra al usuario final, se utilizó el framework Angular en su versión 10. En la Figura 6, se puede observar un resumen de la arquitectura que se detalla más adelante.

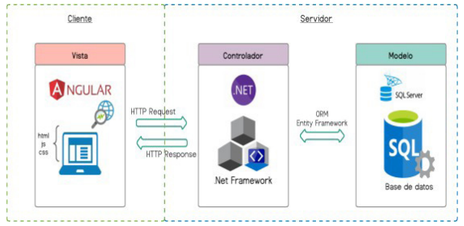

Fig. 6. Arquitectura MVC del sistema

B. MODELO

Para la implementación del modelo, se utilizó un ORM (Object Relational Mapping) llamado Entity Framework que es compatible con el ambiente de trabajo utilizado (.NET). Este ORM permite utilizar objetos de clase específicos para procesar datos sin tener que prestar atención a las tablas y columnas de la base de datos donde se almacenan.

Para este proyecto se trabajó con el gestor de base de datos SQL Server. En la Figura 7, se puede observar cierta parte del modelo de datos la cual representa la estructura de un proceso electoral.

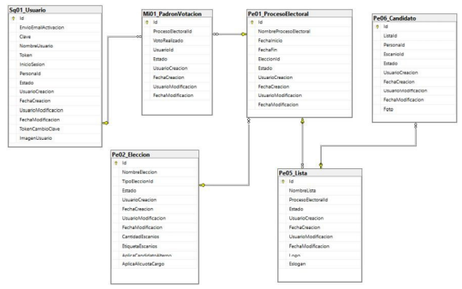

Fig. 7. Modelo físico de la base de datos

C. VISTA

Para la implementación de las vistas del aplicativo web se ha utilizado Angular como framework de desarrollo. Con esta herramienta se puede separar en archivos la parte del código, HTML, y estilos. El sistema cuenta con una pantalla de Login para el inicio de sesión de los usuarios del sistema. Una vez que el usuario inicia sesión exitosamente podrá observar la pantalla de bienvenida que junto con las demás pantallas tienen en común un menú lateral en la parte izquierda donde se encuentra la imagen del usuario, el nombre y las rutas de menú permitidas para cada usuario. También cuenta con una cabecera en donde está la imagen del usuario y un botón para poder abrir o cerrar el menú, y, por último, en el centro se tiene un área que irá cambiando, dependiendo de la ruta a la que se dirige el usuario.

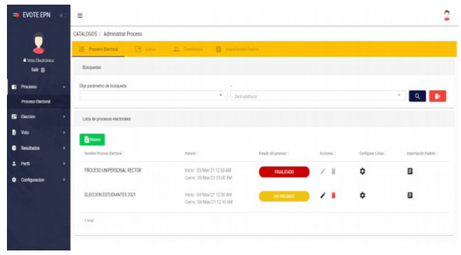

Fig. 8. Pantalla de configuración de procesos.

D. CONTROLADOR

El controlador de la aplicación se ha implementado bajo el framework de .Net y con el lenguaje de programación C#.

Con estas herramientas se ha desarrollado el backend a manera de capas. La capa más interna es la de acceso a datos, en la cual se tienen las entidades y los diferentes métodos para acceder a la base de datos. Cabe mencionar que para las funciones de acceso a datos se ha implementado el patrón repositorio. Con este patrón se crea una sola vez cada método de acceso para que puedan ser utilizados por cualquier entidad.

Luego, se encuentra la capa de lógica de negocio, que viene a ser la capa intermedia entre la capa web API y el acceso a datos. En esta capa, se han implementado todas las reglas de negocio, operaciones con entidades, validaciones, consultas etc. Como resultado se tiene un conjunto de servicios que son expuestos a las capas superiores.

La capa web API es la encargada de recibir las peticiones http y comunicarse con algún servicio para devolver la información requerida por el lado del cliente. En esta capa, se han implementado técnicas de seguridad como la autenticación por token, es decir, si una petición no tiene un token válido, generado por el sistema, no se permite el paso a una capa más interna y se retorna una respuesta de acceso no autorizado. Un ejemplo de controlador a nivel de esta capa se lo puede observar en la Figura 9.

```
[HttpPost]
[Route("VESLPrcCtv")]
public HttpResponseMessage VESLPrcCtv()
{
    try
    {
        var token = Request.Headers.Authorization.Parameter;
        _tokenValidator.ValidarToken(token);
        return Request.CreateResponse(HttpStatusCode.OK, _procesoElectoralService.ObtenerTodosProcesosElectoralesActivos());
    }
    catch (Exception ex)
    {
        return Request.CreateResponse(HttpStatusCode.BadRequest, _apiResponseMessage.crearOtoErrorExceptionMessage(ex));
    }
}
```

Fig. 9. Servicio de obtención de procesos

E. GENERACIÓN DEL VOTO

Una vez que el elector ha seleccionado sus opciones, en caso de que el voto sea válido, se tiene un conjunto de datos de tipo Json con la información necesaria para poder guardar el voto. En caso de que los votos sean de tipo nulo o blanco, el voto se tratará como un arreglo vacío. Después de que el elector verifica su voto y da clic en el botón "Emitir voto", la aplicación web en el lado del cliente procede con el cifrado del mismo para posteriormente enviarlo.

Para el cifrado, lo primero que se hace es generar una llave simétrica que será utilizada para encriptar el voto con el algoritmo AES. Para realizar este cifrado se ha utilizado la biblioteca CryptoJS la cual abarca una gran colección de algoritmos criptográficos estándar y seguros. Por temas de implementación se ha elegido generar también un vector de inicialización (IV) de 16 bytes junto con la llave (keyAES) de 32 bytes. Para la generación de estas llaves se ha desarrollado en el sistema una función, la cual recibe una longitud de cadena y selecciona de forma aleatoria carácter por carácter, de entre un conjunto de caracteres alfanuméricos, para obtener cada llave.

Una vez que tiene el vector de inicialización y la llave AES se las envía junto con el voto en texto plano a la función de encriptación. En esta función, se convierten el par de llaves a formato UTF8 ya que es un requisito de la librería. Luego se hace uso de la función "CryptoJS.AES.encrypt()" adjuntando el par de llaves ya convertidas y el voto. Esta función retornará el voto ya cifrado con el algoritmo AES. En la Figura 10 se detalla dicha función.

ISSN:1390-9266 e-ISSN:1390-9134 LAJC 2022
DOI: 10.5281/zenodo.6762958

```
encriptarVotoConAes(voto, key,iv) {
        var votoString = JSON.stringify(voto);
        var keyUTF8 = CryptoJS.enc.Utf8.parse(key);
        var ivUTF8 = CryptoJS.enc.Utf8.parse(iv);
        var cifradoAES = CryptoJS.AES.encrypt(CryptoJS.enc.Utf8.parse(votoString), keyUTF8, {
            iv: ivUTF8,
            mode: CryptoJS.mode.CBC,
            padding: CryptoJS.pad.Pkcs7
        });

        return cifradoAES;
    }
```

Fig. 10. Encriptación de voto con AES

Siguiendo la arquitectura de voto propuesto, una vez que se tiene el voto cifrado se procede con la generación de una máscara única para cada voto. Esta máscara ayudará para posteriormente guardar el voto sin tener que asociarlo directamente con el elector. Además, el elector será el único conocedor de su máscara con la cual también podrá comprobar la integridad de su voto.

Para la generación de la máscara se ha usado una función de hash (SHA512) aplicada al voto cifrado y añadiendo ciertos parámetros para brindar aleatoriedad. La estructura de la máscara es SHA-512(t1 | votoAES | t2), donde:

- t1: Instante de tiempo 1 en el cual empieza el proceso de creación de la máscara.
- t2: Instante de tiempo 2 más un valor generado aleatoriamente.
- votoAES: Es el voto cifrado con AES.

F. AUTORIZACIÓN Y FIRMA CIEGA

Una vez que se tiene el voto cifrado y la máscara se procede con el proceso de identificación, autorización y firma ciega del voto. En este instante se realiza la comunicación con el esquema de identificación para lo cual se envía el token de identificación del usuario elector junto el voto cifrado, la máscara generada y un identificador del proceso electoral, codificado en Base64, al cual se desea hacer referencia para el voto.

En el lado del servidor primero se verifica que el usuario conste en el padrón electoral del proceso al cual se está haciendo referencia, que no haya votado aún, que se encuentre dentro del horario habilitado. Si cumple con los requisitos impuestos se autoriza el voto al elector y empieza el proceso de firmado ciego.

Para implementar el proceso de firmado ciego se ha hecho uso de un proveedor de servicios criptográficos "System.Security. Cryptography" desarrollado en C# y que sirve para .Net. Este servicio necesita en primer lugar un par de claves, la pública y la privada, para poder trabajar con el algoritmo RSA.

Una vez con las claves ya generadas, para la firma del voto se utiliza solo la clave privada. Haciendo uso de la biblioteca de criptografía se utiliza la función SignData(); a la cual se le pasa el String cifrado (AES) del voto concatenado con la máscara y codificados en formato UTF8. Luego se especifica el tipo de función hash que se desea utilizar, en este caso SHA512. Esta función devuelve como resultado la firma digital de los datos proporcionada.

```
public string FirmaDigital(string text)
{
    //Transformar el mensage a un array de bits en codificación UTF8
    var encoder = new UTF8Encoding();
    var bytesText = encoder.GetBytes(text);
    // Hash y firmar los datos.
    var signedData = _privateKey.SignData(bytesText, CryptoConfig.MapNameToOID("SHA512"));
    return Convert.ToBase64String(signedData);
}
```

Fig. 11. Servicio de firma digital

G. GUARDADO DEL VOTO

En el lado del cliente cuando ya se ha obtenido la firma digital se procede con el guardado del voto. Para esto se envían la firma digital, el voto cifrado con AES, las respectivas llaves para el descifrado y la máscara. Como se aprecia en esta parte dejamos de lado el token identificador del elector, garantizando el voto secreto, ya que si se cuenta con una firma digital significa que el usuario ha sido autenticado y autorizado.

El esquema o mesa de voto recibe toda esta información necesaria para primero, poder comprobar la validez de la firma. De la misma manera, en este punto se utiliza la función de la librería criptográfica VerifyData(); la cual utiliza la clave pública del esquema de identificación junto con los datos firmados y los datos de origen para verificar la firma.

```
public bool VerificarFirmaDigital(string datosOriginales, string datosFirmados)
{
    var encoder = new UTF8Encoding();
    var datosOriginalesBytes = encoder.GetBytes(datosOriginales);
    var datosFirmadosBytes = Convert.FromBase64String(datosFirmados);

    var decryptedBytes = _publicKey.VerifyData(datosOriginalesBytes, CryptoConfig.MapNameToOID("SHA512"), datosFirmadosBytes);

    return decryptedBytes;
}
```

Fig. 12. Función de verificación de firma digital

Si la verificación es exitosa significa que tanto el voto como la máscara son correctos y no han sido manipulados por lo tanto se procede con el guardado del voto. Se debe recordar que en este punto, el voto que se ha recibido por parte del elector sigue estando cifrado a diferencia que esta vez ya se conoce el valor de la clave y el vector de inicialización. Para la obtención del texto descifrado se ha creado una función la cual recibe los valores de las claves AES y el voto cifrado y devuelve la lista de opciones

ISSN:1390-9266 e-ISSN:1390-9134 LAJC 2022
DOI: 10.5281/zenodo.6762958

elegidas por el elector en forma de lista de objetos.

Finalmente, con la lista de opciones ya descifradas más los datos proporcionados por el elector se procede a guardar la información en la base de datos. Para soportar este proceso se crearon dos tablas: la tabla voto y la tabla opción. Si el voto a guardar es un voto válido se guarda primero la máscara y el estado en la tabla voto y se crea un registro por cada opción seleccionada guardándolo en la tabla opción. En caso de que el voto sea de tipo nulo o blanco solamente se guarda un registro en la tabla voto junto con el estado correspondiente.

IV. ANÁLISIS DE RESULTADOS

Luego de haber realizado las pruebas de funcionalidad, para un total de 34 historias de usuario, se pudo constatar que el sistema superó satisfactoriamente cada caso de prueba. Por lo tanto, el sistema presenta, según los requerimientos solicitados, una funcionalidad y completitud del 100 %.

En cuanto a la usabilidad del sistema, luego de realizadas las respectivas encuestas a los usuarios, y realizado el cálculo de los resultados obtenidos mediante la herramienta metodológica SUS; se obtiene una puntuación de 84,5/100. Basándonos en la escala proporcionada para esta metodología, la cual se puede observar en la Figura 13, se concluye que el sistema es excelentemente usable.

Fig. 13. Puntuación de aceptabilidad del SUS

V. CONCLUSIONES

A. CONCLUSIONES

- Se ha desarrollado con éxito un aplicativo web que ayuda a automatizar los procesos electorales de la Escuela Politécnica Nacional. El aplicativo ha sido desarrollado con el objetivo de ser configurable y soportar todos los posibles tipos de procesos electorales que se manejen en la institución, incluyendo las elecciones de Rector y sus respectivos Vicerrectores, elecciones internas de las diferentes facultades, elecciones de representantes ante los diferentes consejos, entre otras.
- Después de analizar los requerimientos obtenidos, se definieron cuatro módulos principales para el sistema: Configuración, Procesos, Elecciones y Resultados, mismos que sirvieron para agrupar las funcionalidades y tener un menú de usuario más organizado. Además, esta modularización ayudó a priorizar de mejor manera los requisitos a desarrollar, facilitando así su planificación.
- Gracias al concepto de firma ciega implementado bajo el algoritmo de RSA se logró crear un esquema para la firma digital de los votos de los electores, garantizando la integridad y el anonimato del voto. Además, gracias a esta implementación se pudo conocer más en profundidad sistemas criptográficos como RSA, AES, SHA, firma digital.
- Mediante pruebas de funcionalidad, se determinó que el sistema se encuentra completamente funcional y cumple satisfactoriamente la totalidad de sus requisitos. Además, del resultado obtenido mediante el cálculo de la usabilidad especificado en la metodología SUS, se concluye que el sistema tiene una facilidad de uso excelentemente aceptable.
- El sistema de votación electrónica fue desplegado en un hosting público para llevar a cabo una simulación de un proceso electoral dentro de la EPN y orientado a un número considerable de usuarios. Mediante la aplicación de encuestas, tras finalizar el proceso electoral, se pudo determinar que los usuarios tuvieron una gran acogida al sistema. Esto se debe a la simplicidad de uso y al diseño amigable de la aplicación web.

B. RECOMENDACIONES

- Para la implementación del sistema se ha desarrollado una arquitectura de voto, en donde se separan mediante esquemas la mesa de identificación de la mesa de votación, aunque, compartiendo recursos como la base datos. Por lo tanto, para una mejora en la implementación del sistema de voto electrónico se recomienda separar la funcionalidad de la mesa de votación en una entidad independiente, tanto en infraestructura como en lógica de negocio. Esto ayudaría a fortalecer el anonimato de los electores, ya que dicha entidad se encargaría únicamente de validar y

almacenar los votos firmados, dejando de lado cualquier relación con los usuarios del sistema.

- El sistema desarrollado se enfocó en ofrecer seguridad a nivel del aplicativo web, asumiendo tanto la seguridad de la base de datos como una infraestructura de comunicación altamente disponible. Por tanto, se marca como pendiente la implementación de comunicaciones redundantes, para aumentar la disponibilidad del sistema y reducir la probabilidad de fallos por errores de infraestructura. Además, al ser un sistema que almacena datos muy sensibles, se recomienda manejar una auditoria a nivel de gestor de base de datos, para evitar posibles intervenciones o manipulación de la información. Con estas mejoras, se espera tener un sistema sumamente seguro en todos los ámbitos.

- Para poder guardar el voto de un elector, se utilizó un método que genera una máscara única a partir de dicho voto. Se realizaron alrededor de diez mil pruebas del método de generación de la máscara, con un mismo voto y cambiando únicamente los instantes uno y dos, teniendo como resultado que en ningún momento se repitió dicha mascara. Aun así, se recomienda la implementación de un mecanismo de vuelta atrás, en donde si una máscara generada se repite, se vuelva al esquema de identificación y se genere otra máscara hasta conseguir una que sea única.

REFERENCIAS

[1] L. Borja y D. Rodriguez, «Propuesta Técnológica para la Sistematización del Proceso de Voto Electoral Estudiante de la Unidad Educativa Particular Dante Alighieri del Distrito 3 de la Ciudad de Guayaquil,» Repositorio Universidad de Guayaquil, 2016. [En línea]. Available: https://rraae.cedia.edu.ec/Record/UG_9c115852f00593e2b70a007749e4bdd7. [Último acceso: 2022].

[2] M. Bautista, A. Martínez y R. Hiracheta, «El Uso de Material Didáctico y las Tecnologías de Información y Comunicación (TIC's) para Mejorar el Alcance Académico,» Ciencia y Tecnología, vol. 14, nº ISSN 1850-0870, pp. 183-194, 2014.

[3] A. Bates y A. Sangra, «La Gestión de la Técnología en la Educación Superior: Estrategias para Transformar la enseñanza y el Aprendizaje,» Barcelona, John Wiley & Sons International Rights, Inc, 2016, p. 15.

[4] Consejo Politécnico, Reglamento General de Elecciones de la Escuela Politécnica Nacional, 2019.

[5] Escuela Politécnica Nacional, Estatuto de la Escuela Politécnica Nacional, 2019.

[6] ISO, La Guía Internacional ISO/TS 54001:2019: Requerimientos Específicos para la Aplicación de la Norma ISO 9001:2015 a Organizaciones Electorales en todos los Niveles de Gobierno, 2019.

[7] Proyectos Ágiles, «Beneficios de Scrum,» [En línea].Available: https://proyectosagiles.org/beneficios-de-scrum/. [Último acceso: 2022].

[8] Scrum, «La Guía de Scrum,» 2013.

[9] L. Cuya, «Aplicación de la Metodología Ágil Scrum en el Desarrollo de un Aplicativo para la Gestión de Observadores Electorales,» 2015. [En línea]. Available: http://repositorio.untels.edu.pe/jspui/bitstream/123456789/322/1/Cuya_Liliana_Trabajo_de_Investigacion_2014.pdf.

[10] MVC, «Servicio de Informática ASP.Net MCV 3 Framework,» [En línea]. Available: https://si.ua.es/es/documentacion/asp-net-mvc-3/1-dia/modelo-%20vistacontrolador-mvc.html.

[11] D. Chaum, «Blind signatures for untraceable payments,» Advances in cryptology, pp. 199-203, 1983.

[12] C. Moreno, «Diseño e implementación de un sistema de voto electrónico,» 2016.

[13] I. Sommerville, Ingeniería del software, Madrid: Pearson Education S.A, 2005.

ISSN:1390-9266 e-ISSN:1390-9134 LAJC 2022
DOI: 10.5281/zenodo.6762958

AUTHORS

Jose Azadobay

JOSE DANIEL AZADOBAY CUTIOPALA nació en Quito - Ecuador, el 14 de enero de 1996. Se graduó como bachiller en Físico Matemático en la “Colegio Experimental Juan Montalvo”, Quito - Ecuador, en el año 2013. En 2021 se graduó de la Escuela Politécnica Nacional, Facultad de Ingeniería de Sistemas obteniendo el título de “Ingeniero en sistemas informáticos y ciencias de la computación”,. Ingeniero de software, proactivo, líder, autodidacta con experiencia en React JS, Angular, .Net, Nest JS, Django, SQL, AWS de aprendizaje rápido y continuo, apasionado por el desarrollo web y todo lo referente a la industria. Durante su experiencia profesional ha trabajado como desarrollador de software destacándose y dominando varios lenguajes como C#, Python, Java, Typescript, Javascript, siendo este último en particular su fuerte. Actualmente trabaja para la representación de Hyundai en Ecuador con el cargo de desarrollador Full Stack, liderando varios proyectos e innovando en tecnologías actuales.

Michael Morales

Estudiante graduado de la facultad de ingeniería de Sistemas de la Escuela Politécnica Nacional en agosto del año 2020. Nacido en Quito – Ecuador en el seno de una familia de clase media. Entre sus mayores intereses profesionales se encuentran el desarrollo de aplicativos web y móviles, enfocado tanto en frontend como en backend y en el diseño de aplicaciones con gran énfasis en la experiencia del usuario final, usabilidad y tendencias de interfaces amigables e inclusivas. Siempre motivado por el avance tecnológico y siempre a la vanguardia de los nuevos paradigmas de programación y desarrollo de software. Practicante incondicional de metodologías de desarrollo ágiles, especialmente Scrum y XP. Padre de un niño y esposo. Amante de los deportes de esfuerzo corporal y de los instrumentos musicales de cuerda y viento. Lector de novelas de Ficción, autobiografías y libros con mensajes protesta, aficionado de las obras del escritor Gabriel García Márquez.

ISSN:1390-9266 e-ISSN:1390-9134 LAJC 2022
DOI: 10.5281/zenodo.6762958

Hernán Ordoñez

Ingeniero en Sistemas Informáticos y de Computación, en 2014, y Magíster en Software Mención Calidad, en 2018, por la Escuela Politécnica Nacional, Quito, Ecuador. Por varios años docente de la Escuela de Formación de Tecnólogos (ESFOT) de la EPN, y en la actualidad es profesor de la Facultad de Ingeniería de Sistemas (FIS), en la Escuela Politécnica Nacional. Autor y coautor de un sin número de publicaciones de interés orientados a aspectos tecnológicos. Sus intereses de investigación incluyen la creación y gestión de software, y la calidad del software. Practicante incondicional de deportes como el futbol, miembro activo del grupo de seleccionados de la facultad de Ingeniería en Sistemas de la EPN.

Carlos Montenegro

Obtuvo su título de MSc en Informática y Ciencias de la Computación en 2001. Actualmente, es profesor en la Escuela Politécnica Nacional. Además, se desempeña como decano de la facultad de Ingeniería de Sistemas de dicha institución y como CEO del departamento de Ciencias de la computación. También ha sido testigo experto de varios incidentes de seguridad nacional. Entre sus intereses académicos se encuentra el aprendizaje de máquina, minería de datos, inteligencia artificial y la gestión de las TIC. Montenegro es autor y coautor de más de doce publicaciones en sus áreas de interés.