UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escuela Técnica Superior de Ingeniería Informática

Verificación automática de un protocolo de voto electrónico.

Trabajo Fin de Máster

Máster Universitario en Ciberseguridad y Ciberinteligencia

AUTOR/A: Acedo Arroyo, Josep Vicent

Tutor/a: López Rodríguez, Damián

Cotutor/a: Escobar Román, Santiago

Director/a Experimental: APARICIO SANCHEZ, DAMIAN

CURSO ACADÉMICO: 2021/2022

Resumen

La seguridad y transparencia de los procesos electores son la garantía fundamental de la voluntad popular. Sin ellos es imposible la construcción y el mantenimiento de la democracia. Las nuevas tecnologías van integrándose en los procesos electorales lentamente, pero aún hay muchas resistencias y problemas que resolver. La propuesta de protocolos de voto electrónico ha de ir acompañada de métodos formales para la validación de estos protocolos. El protocolo TAVS es una aproximación elegante, sencilla y flexible para la implementación del voto remoto. Maude-NPA es una herramienta formal plástica y potente para poder llevar a cabo estas validaciones. En este trabajo se utilizará la Maude-NPA para hacer una primera aproximación a la validación formal de TAVS, demostrando los beneficios de este tipo de evaluación.

Palabras clave: TAVS, Maude-NPA, voto electrónico, evaluación, automática.

Abstract

The security and transparency of the electoral processes are the fundamental guarantee of the popular will. Without them, the construction and maintenance of democracy is impossible. New technologies are slowly being integrated into electoral processes, but there is still much resistance and problems to be solved. The proposal of electronic voting protocols has been accompanied by formal methods for the validation of these protocols. The TAVS protocol is an elegant, simple, and flexible approach for the implementation of remote voting. Maude-NPA is a plastic and powerful formal tool to be able to carry out these validations. In this work, the Maude-NPA was used to make a first approximation to the formal validation of TAVS, demonstrating the benefits of this type of evaluation.

Keywords : voto electrónico, TAVS, Maude-NPA, verificación automática.

Tabla de contenidos

Tabla de ilustraciones

1. Introducción

Si se preguntase a cualquier individuo, por los ámbitos en los que la informática y las telecomunicaciones son importantes, pocos indicarían los procesos electorales. Es cierto que para poder desarrollar un proceso electoral transparente, eficaz y eficiente es vital disponer de comunicaciones y censos actualizados; y que para ello son trascendentales los sistemas de información. Pero sin querer restarles importancia, estos son transparentes para el votante, quien los desconoce o los da por asumidos.

Esta apreciación surge debido a que estos sistemas de información operan en la trastienda del proceso electoral. El elector desarrolla su rol muy concreta y tradicionalmente: la elección de la papeleta y su introducción en la urna. Acción física, e -sobre todo la última- intransmisible y presencial.

La tecnología solo ha desplazado al voto tradicional muy tímidamente, y aun con desconfianza acerca de su integridad, confidencialidad o -paradójicamente- disponibilidad. Aun así, algunos países han comenzado a considerar o incluso implementar el concepto -véase por ejemplo Wikipedia [1]- con mayor o menor éxito. Las motivaciones a favor son múltiples:

- La deslocalización física del evento de voto supondría mayor disponibilidad, la eliminación de colas de espera y el aumento de la participación, produciendo un robustecimiento general de la democracia.
- La reducción del tiempo y medios necesarios para el conteo y publicación de resultados, dado que incluso en algunas democracias consolidadas puede requerir de varios días.
- Se obtendría una considerable simplificación logística del proceso electoral y su abaratamiento económico. Esto además permitiría ampliar el número de consultas y el alcance a temas de menor relevancia, permitiendo una participación ciudadana más directa en los asuntos de gobierno.

Pero siendo todo proceso electoral un evento con tan graves implicaciones -respeto de los derechos fundamentales, intereses políticos y económicos nacionales o geoestratégicos- la adopción de cualquier sistema de voto electrónico supone buscar la transparencia y seguridad absolutas. Además, un nuevo sistema no puede suponer dificultades añadidas para la participación -nivel de alfabetización tecnológica, disponibilidad personal de medios, etc.

Diversos autores han propuesto protocolos de voto electrónico, todos ellos hacen uso de bases criptografías, como medio para asegurar los componentes[1] necesarios para sustituir al voto tradicional, manteniendo sus características. Entre ellos se encuentra la propuesta de Larriba, Sempere y López en [2], sobre la que se centra este trabajo.

Cualquier propuesta en ciencia ha de ser evaluada y contrastada. Hasta la fecha, algunos de los protocolos han sido valorados con mayor o menor fundamento. Sin

[1] Confidencialidad, integridad y disponibilidad.

embargo, los trabajos de crítica, aun existiendo análisis formales, no habían hecho uso de métodos automáticos estos análisis de seguridad de los protocolos.

La existencia de herramientas de análisis de protocolos como Maude-NPA de Escobar, Meadows y Meseguer en [3] o Tamarin de Basin, Cremers, Dreier y Mei en [4] permiten describir los protocolos y ataques a evaluar; y formalizar y sistematizar esta evaluación. Maude encuentra sistemáticamente el camino de éxito asociado a un ataque, por lo que además constituye una referencia común de trabajo para la comparación tanto de la robustez de protocolos como de la potencia de los ataques.

1.1. Motivación

El Departamento de Sistemas Informáticos y Computación dispuso este trabajo bajo el planteamiento de oportunidad de evaluación de la propuesta de voto electrónico hecha por Larriba, Sempere y López en [2]; buscando las garantías de sistematicidad y formalización que ofrece Maude-NPA de Escobar; Meadows y Meseguer [3]. Un análisis de estas características, aun parcial, que confirme las hipótesis de los autores del *paper* aporta robustez y credibilidad al protocolo diseñado.

Por parte del alumno la motivación ha sido múltiple: Primero, la excelente ocasión de experimentar con la herramienta Maude-NPA en un protocolo real y para el que se dispone de acceso directo a los autores -tanto de la herramienta como del protocolo.

Durante el propio primer curso de este Máster se abordó el impacto que los sistemas de evaluación automáticos tienen sobre sistemas reales, como en la página de la Cancillería Federal Suiza [5] donde se puede encontrar el documento [6] de valoración de sus sistemas de voto electrónico.

Al conocer de la existencia de la herramienta Maude-NPA, el alumno razonó sobre el enorme potencial del análisis de protocolos -especialmente criptográficos- dentro del ámbito militar en el que desarrolla su trabajo. Es natural que al disponer de la coyuntura apropiada para iniciarse en el conocimiento de esta herramienta, la elección estuviese servida.

Pero además de lo anterior, y a título más personal; hay que considerar el impacto que podría representar disponer de un protocolo de voto electrónico fiable y validado con el que desarrollar los procesos electorales. El alcance de un desarrollo de estas características transciende lo técnico y se transforma en un cambio social, con una repercusión global. ¿Cómo no participar de estar en la primera fila de algo así?

1.2. Objetivos

Los objetivos que se plantean en este trabajo corresponden a los iniciales en la validación del protocolo. La validación total de TAVS requería de análisis en profundidad, pero en este trabajo en concreto se trabaja sobre los prerrequisitos, para confirmar el peso que estos tienen en los aspectos de confidencialidad y autenticación del protocolo. Las cualidades únicas que aporta TAVS -donde reside su aportación diferencial respecto a otras propuestas de voto electrónico- habrán de evaluarse en siguientes trabajos.

Entre estos prerrequisitos, TAVS, considera que los canales de comunicación son seguros. En esta primera tarea se evaluarán dos escenarios: primero aquel en el que los canales no son seguros, y posteriormente el escenario en el que lo son. Es decir, evaluar la importancia del prerrequisito de confidencialidad.

En una segunda tarea, se pretende evaluar limitadamente -por la indicada asociación a los prerrequisitos - la capacidad de autenticación del protocolo, para asegurar la integridad/autenticación del proceso. Es decir, demostrar el valor de los prerrequisitos ante los intentos de suplantación del Elector o la Autoridad de Identificación.

Con ambas tareas se buscará reafirmar la validez de la propuesta hecha por los autores del paper y el valor del planteamiento de los prerrequisitos.

1.3. Impacto esperado

Las ventajas que este trabajo pretende aportar son la validación inicial del protocolo TAVS, que permita asentar este para más tarde evaluar las cualidades diferenciales al proceso electoral que aporta -transparencia, unicidad, elegibilidad, privacidad libertad ante coerción, negación plausible, etc.

Paralelamente, un resultado positivo de este trabajo supondría un paso pequeño paso valioso dentro de los necesarios para que TAVS incida en los objetivos de desarrollo sostenible; en concreto para el décimo sexto objetivo de desarrollo: «Paz, justicia e instituciones sólidas».

1.4. Metodología

Maude-NPA es una herramienta que realiza análisis de alcanzabilidad simbólica hacia atrás. Partiendo del estado final al que deseamos llegar, genera todos los caminos posibles y los evalúa, descartando los no útiles hasta llegar a una situación de partida realista. Lo que se obtiene es todo el camino desde una situación de partida, los pasos que el atacante debe realizar, y el estado final.

La metodología de trabajo consistirá en la demostración del caso base de funcionamiento del protocolo y la validación de tres escenarios. Esos escenarios se codificarán en su estado final y buscará soluciones automáticamente.

En un primer escenario se demostrará la validez de los prerrequisitos respecto de la confidencialidad o secreto del sentido de voto. Se planteará un mismo ataque a la confidencialidad, contra ambas variantes: medio inseguro, y medio seguro, dejando a Maude-NPA la búsqueda de solución. Es importante remarcar que el resultado será extrapolable a los demás escenarios.

Para los escenarios secundarios se tratará de demostrar -solo bajo relajación de requisitos- la debilidad ante la manipulación del voto, respecto del Elector y de la Autoridad de Identificación. Aplicando el resultado del primer escenario, se contrarrestarán estos y se aceptará la robustez del protocolo.

1.5. Estructura

Este trabajo se divide en ocho capítulos principales. El primero corresponde a la introducción, motivación, objetivos, impacto esperado y metodología. El segundo se centra en el estado del arte en cuanto a propuestas de voto electrónico y su evaluación de seguridad. El tercero presenta el protocolo a ser evaluado: TAVS. El cuarto Introduce el modelado en Maude-NPA. El quinto desarrolla el análisis del protocolo TAVS con Maude-NPA. El sexto presenta los experimentos realizados para los cuatro supuestos. El séptimo aporta las conclusiones. Por fin el octavo indica futuros trabajos a realizar sobre este tema particular. Hay dos capítulos adicionales, más administrativos, octavo y noveno que comprenden la bibliografía y el glosario de términos específicos.

1.6. Convenciones

En esta obra, se seguirán las siguientes convenciones:

- El pseudo código se muestra en Times New Roman. Sus variables relevantes se remarcarán en cursiva.
- El código fuente se muestra en *Times New Roman* cursiva.
- Los resultados de las herramientas se presentan a través de capturas de pantalla; sin detrimento de posteriores explicaciones.
- Las palabras extranjeras se remarcarán en cursiva.
- Se entrecomillarán las citas textuales externas a la obra.
- Se citarán las referencias de las que se haya extraído la información según ISO-690 numérica anteponiendo el apellido de los autores. En el punto noveno del documento se encuentra la tabla de referencias.

2. Estado del arte

A pesar de que algunos países e instituciones ya han dado el primer paso en la realización de elecciones mediante sistemas electrónicos, el concepto está lejos de estar completamente resuelto. Hay un vivo interés académico por resolver los problemas pendientes y conseguir madurar la solución al voto electrónico.

Los modelos de voto por correo basados en DRE[2] se quedan atrás respecto del voto remoto y tienen importantes taras, como plantean Bannet, Price, Rudys, Singer y Wallach en [7] o Schneier en [8].

Para los sistemas en remoto -pero también en general- el principal problema es conseguir el anonimato del elector manteniendo la confidencialidad; protegiendo a este de la coerción. Una buena definición del problema de la coerción y algunas propuestas de solución se puede encontrar en [9] de Juels, Catalano y Jakobsson.

TAVS de Larriba, Sempere, Jose y Lopez [2] es una excelente solución, robusta y sencilla de implementar, a evaluar en este trabajo. Entre la oferta académica de otras soluciones al problema tenemos Acquisti en [10], que también propone el uso de propiedades homomórficas de algunos criptosistemas para desacoplar identidad y sentido de voto. Hay otras aproximaciones interesantes como las presentada por Ryan, Peter, Bismark, Heather, Schneider y Xia en [11] basada en la aleatorización y encriptado del orden de candidatos previo al evento de voto, incluyendo la información en el propio voto cifrado. Los autores Bohli, Mueller-Quade y Roehrich en [12] proponen utilizar la aleatorización en el registro del sentido de voto, pero requiere de generadores de números aleatorios confiables en cabinas físicas. Otra referencia de interés basada en demostración de teoremas es la ofrecida por Miramirkhani, Jalili, y Yarmohamadi en [13].

Otra área por considerar es la evaluación de protocolos criptográficos. Podemos encontrar un interesante resumen del estado en Cortier [14]. Es posible retroceder hasta referencias tan tempranas como Blanchet [15] o Chiara, Mikael, Pierpaolo, Flemming y Riis [16], pero si nos interesan las referencias más cercanas y sobre sistemas de voto podemos que recurrir a Cortier, Eigner, Kremer, Maffei, y Wiedling [17] como sistema general de validación, a Delaune, Kremer y Ryan [18] con aplicación del cálculo pi, o Kremer y Ryan en [19]. Un buen ejemplo de trabajo práctico utilizado como base para este trabajo ha sido el trabajo de Palop [20].

En cuanto a las herramientas que están disponibles, podemos referenciar -Maude-NPA aparte- a Tamarin Basin, Cremers, Dreier y Mei [4], que ya fue utilizado parcialmente en Palop [20]. Para la validación basada en cálculo-pi está disponible Blanchet y Cheval [21].

[2] Direct Recording Electronic.

3. Protocolo de voto TAVS

TAVS es el acrónimo utilizado para referirse al protocolo presentado en Larriba, Sempere y López [2]. En dicho artículo los autores proponen un elegante protocolo de voto electrónico multiautoridad basado en firmas ciegas. A diferencia de otras propuestas, este protocolo se basa en la sencillez y en la asunción de no relación entre las autoridades.

Las propiedades necesarias del voto electrónico son garantizadas por TAVS bajo unas condiciones concretas. Dichas propiedades son:

- Imposibilidad de relacionar al elector con una papeleta concreta (privacidad)
- Incapacidad de ningún participante para modificar una papeleta sin ser detectada (integridad)
- El sistema solo contabilizará en su recuento final las papeletas formalmente válidas (corrección)
- Cualquier elector censado puede comprobar que su voto ha sido considerado (verificabilidad)

TAVS se basa en la existencia de dos autoridades independientes. La *Identity Authority*[3] (IA) comprueba la identidad de los electores en el censo y certifica su *Pre-Ballot*[4]; el Re*mote Polling Station*[5] (RPS) recibe el *Ballot*[6] certificada, y contabiliza el voto. El tercer participante es el *Elector*, que es quien se relaciona con ambas directamente para certificar su papeleta, y entregarla. Para asegurar la democracia y la verificación se utilizan dos paneles a lo largo del procedimiento: el *Revoked Board*[7] (RB) y el *Public Bulletin Board*[8] *(PBB)*.

El protocolo difiere de otros en que no asume la virtud de las autoridades, y que proporciona métodos para verificar la no manipulación del voto tras cada paso. El elector es capaz de comprobar la honestidad de las autoridades.

La certificación de la papeleta se hace gracias a las propiedades homomórficas de las operaciones a utilizar. En el terreno de la criptografía, las operaciones homomórficas permiten trabajar con textos cifrados de igual manera que si estuvieran en claro, sin necesidad de descifrarlos. TAVS, se beneficia de las propiedades homomórficas sobre lo que se denomina firmas ciegas para permitir el refrendo sobre los mensajes ya cifrados.

[3] Autoridad de Identificación
[4] Pre-papeleta
[5] Colegio Electoral Remoto
[6] Papeleta
[7] Panel de Revocación
[8] Panel Público de Notificación

El protocolo asume ciertos prerrequisitos necesarios para su buen funcionamiento, a saber:

- No considera la codificación de los votos. Es independiente de esta.
- Supone los métodos y procedimientos asociados seguros.
- Asume la capacidad para identificar correctamente a cada votante.
- Admite los medios de comunicación como seguro.
- TAVS asume el uso del algoritmo RSA de Rivest, Shamir y Adleman [22] para la implementación de las firmas.

El segundo punto será de especial interés en la evaluación, pues se verá como el cambio de esta consideración sí afecta a la seguridad.

En términos generales TAVS funciona de la siguiente manera. Aunque cuando entremos en más detalle habrá pasos que no sean considerados por qué no aportan valor en el contexto de esta evaluación del protocolo.

El elector codifica su voto, lo enmascara y ensambla la pre-papeleta utilizando el hash de lo anterior y la operación modular para evitar manipulaciones; utiliza la firma pública de la IA para hacerle llegar la pre-papeleta de forma segura.

La IA recibe la pre-papeleta, comprueba la identidad del elector y que no haya recibido otra petición anterior, en tal caso valida la papeleta firmándola ciegamente con su clave privada del RSA. Este paso gracias al homomorfismo traslada la protección del cifrado dentro del bloque de la papeleta, como veremos más adelante con más detalle. La IA además anota la papeleta en el *Revoked Board* de manera preventiva hasta que elector la confirme.

El elector recibe y confirma la papeleta, notificándolo a la IA; lo que permite eliminar su entrada en el *Revoked Board*.

En el último paso, el *elector* adjunta la máscara en claro a la papeleta y envía todo al *Colegio Electoral Remoto*. Este lo recibe y utiliza la máscara para obtener la inversa y despejar del conjunto el voto y el hash. Luego descifra estos con la clave pública de la IA. Separa ambos campos y comprueba el voto con el hash. Finalmente contabiliza el voto en el *Public Bulletin Board* de manera que el elector pueda comprobar que su voto ha sido considerado como debe. Todo de manera teóricamente anónima.

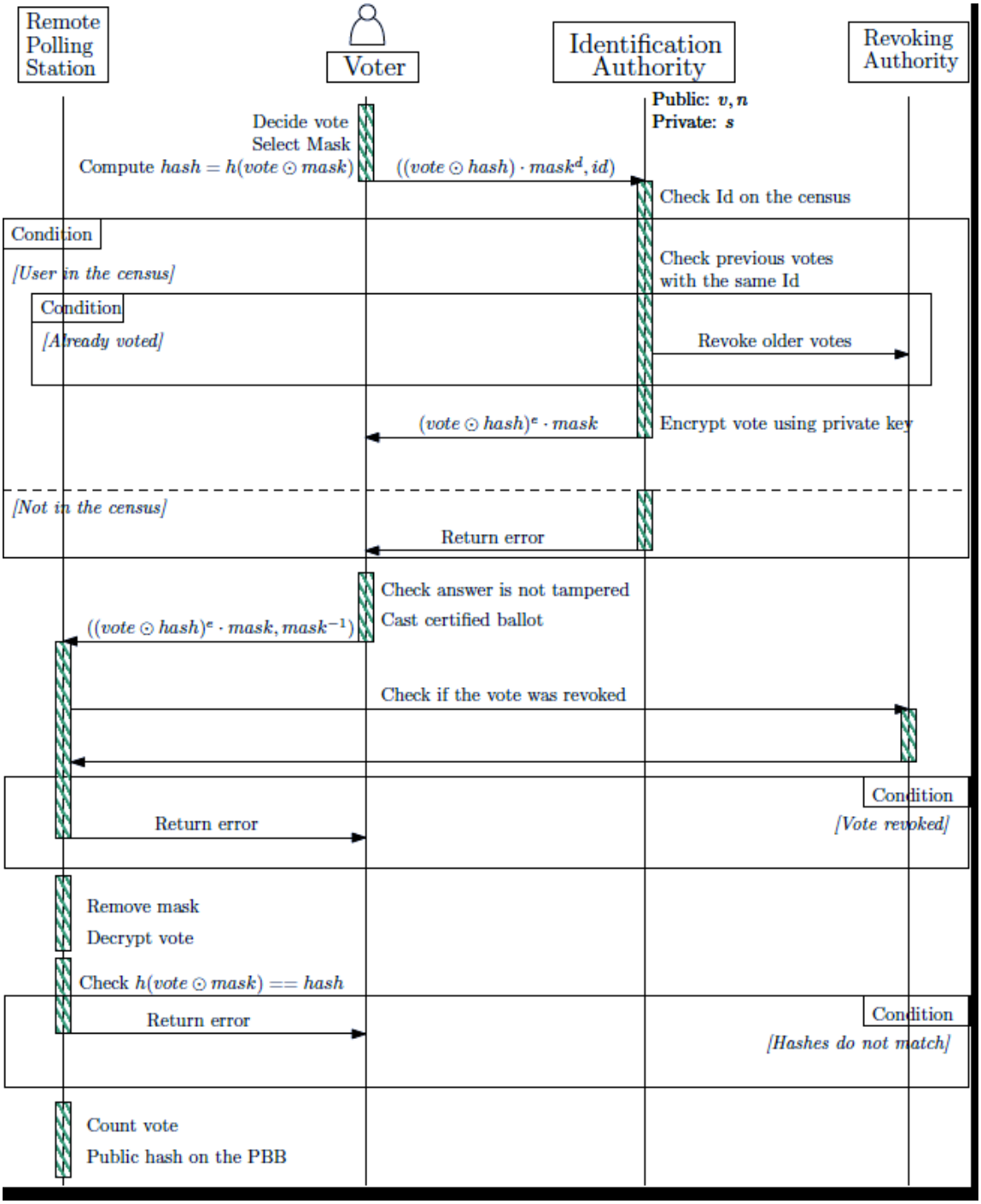

Ilustración 1. Vista general del protocolo TAVS. Extraído de Larriba, Sempere y López [2].

Previamente a las elecciones, la IA habrá generado un par de claves asimétricas y, difundido la componente pública de estas. Esto es preciso puesto que la IA certificará las papeletas con la componente privada y elector y colegio electoral usarán la parte pública para verificar la validez de la certificación. Es este paso previo el que permite establecer y asegurar el pre-requisito de canales de comunicación segura. Solo después de este paso se puede iniciar el funcionamiento de TAVS tras la decisión y codificación del voto por parte del elector.

Volviendo al protocolo de nuestra atención. Un elector generará lo que se denomina pre-papeleta -pues no está validada- de manera independiente. Para ello utilizará el siguiente algoritmo:

Algoritmo 1 *generacion pre-ballot.*
Input: *Componente pública de la clave de firma RSA V IA =< n, v >de la IA*
Input: *Una función hash h de T H bits.*
Output: *Una pre-ballot lista para ser firmada (ciegamente) por la IA*

*1:**Método***
2: Sean T S el número de bits necsarios para codificar n .
3: Sea choice los T S − T H − 1 bits que codifican la elección del candidato.
4: Sea 1 < mask < n un valor (privado) generado aleatoriamente tal que mcd(mask, n) = 1
5: Calcular mask −1 mod n
6: Obtener hash = h (choice mask)
7: preballot = (choice hash) · (mask v mod n) mod n
8: ***retorna*** *< preballot, mask >*
9: ***Fin Método.***

Algoritmo 1. Generación de pre-papeleta.

Los pasos anteriores aseguran que el sentido de la pre-papeleta queda oculto y no puede ser interpretado por nadie más que el propio elector; relaciona el voto con la máscara generada por el elector, lo que previene la manipulación y evita el *duoble vote*.

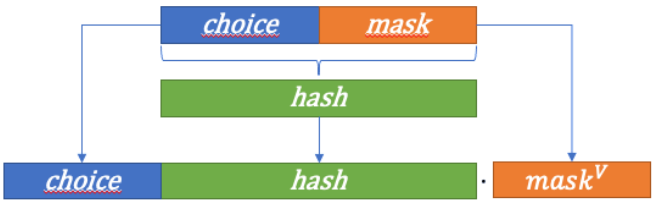

Ilustración 2, Visualización de la estructura y construcción de la pre-papeleta.

Obviamente la componente clave es la máscara: *mask*. Es este el elemento que oculta el sentido del voto y proporciona la propiedad de seguridad buscada. El elector deberá asegurarse de que el secreto del valor de *la máscara* es exclusivamente conocido por él hasta el momento adecuado.

El siguiente paso necesario sería la certificación de la pre-papeleta por parte de la IA. Una vez que el elector tiene su *pre-papeleta* lista, se la hace llegar junto con su identificación a la IA.

Algoritmo 2 *Certificacion pre-ballot.*

Input: *Una pre-ballot pb generada por el elector*
Input: *La identificación del elector*
Input: *Las componentes pública V IA = < n, v >, y privada S IA = < s > de la firma RSA de la IA*

Output: *Un ballot validado (pre-ballot firmado por la IA) o un AvisoDeFalsificación*

1: ***Método***
2: ***si*** *el identificador del elector ya s e encuentra entre los registros de la IA* ***entonces***
3: ***return*** *AvisoDeFalsificación*
4: ***end si***
5: Introducir el identificador del elector en el registro.
6: finCertificación= Falso
7: ***mientras no*** *(finCertificación)* ***hacer***
8: Calcular b = pb s mod n *// la papeleta certificada*
9: Registrar b en el Revocation Board
10: Enviar b al elector y esperar la validación
11: ***si*** *el elector valida b* ***entonces***
12: Eliminar b del Revocation Board
13: finCertificación = Cierto
14: ***end si***
15: ***fin mientras***

16: ***Fin Método***

Algoritmo 2. Certificación de la pre-papeleta.

La IA comprobaría la identidad del elector y si este ya ha votado. Una vez comprobado, firmaría con su componente privada RSA la pre-papeleta, lo anotaría preventivamente en la lista de revocación y lo enviaría al elector, quedando a la espera de que este acepte la papeleta firmada.

Lo más interesante de este paso es la transformación homomórfica que ocurre al firmar la IA con su clave privada, sobre la *pre-papeleta*. Recordemos que la *pre-papeleta* tiene su componente máscara firmada con la clave pública del propio IA. Como puede verse en la ilustración 3, la firma actúa sobre los componentes internos de la *pre-papeleta*, librando la *mask* de su cifrado y trasladándolo a la concatenación del *choice* y *hash*.

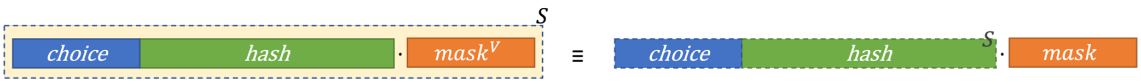

Ilustración 3. Visualización de la transformación homomórfica de la pre-papeleta.

Al recibir la *papeleta*, el elector puede comprobar la corrección y dar el visto bueno a la IA, que extraerá la anotación de la *Revocation Board*. Esta acción convierte la pre-papeleta formalmente en una papeleta utilizable.

Para cerrar el proceso quedaría que el elector enviase la *papeleta* al Colegio Electoral Remoto. El envío consistirá en la *papeleta* certificada por la IA, junto con la inversa de la máscara en claro. Al recibir este paquete, el Colegio Electoral Remoto dispondrá de todas las piezas necesarias para revertir las operaciones hasta obtener el voto.

Con la *papeleta* en su poder, el Colegio Electoral Remoto puede, aplicando la inversa de la *mask*, anular el efecto de la operación multiplicación modular de *mask* sobre la concatenación cifrada de *choice* y *hash*.

Algoritmo 3 Emisión de votos.

Input: Una papeleta certificada *b*
Input: La inversa de la máscara *mask* utilizada en la generación *de la pre-papeleta*.
Input: La componente pública de la clave de firma RSA de la *IA*
Input: La función hash *h usada en el protocolo*.

Método
si *b está en el RevocationBoard* **entonces**
    retorna *AvisoDeFalsificación*
fin si
Calcular $pkg = b \cdot mask^{-1} \bmod n$
Calcular $pkg = pkg^{v} \bmod n$
Hacer *Vote* igual a los $TS - TH - 1$ bits del *pkg*
Hacer *hash* igual a los últimos *TH* bits del *pkg*
si *hash* == *h (Vote mask)* **entonces**
        registrar *Vote como emitido*.
        Publicar *hash* en el *PublicBulletinBoard de las elecciones*
        retorna *Correct*
Si no retorna *AvisoDeFalsificación*
Fin si
Fin Método.

Algoritmo 3. Extracción del voto emitido y publicación.

Gracias a que la clave pública de la IA está disponible para el general, la cifra que protege la concatenación de *choice* y *hash* es reversible. Después trivialmente separará ambos campos y utilizará el *hash* para comprobar la integridad del voto. Como puede verse en la ilustración 4.

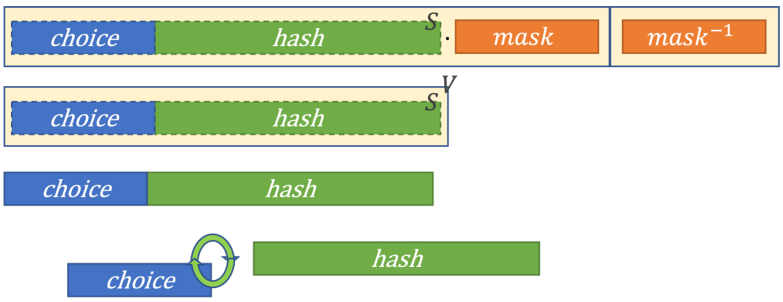

Ilustración 4. Visualización del desensamblado de la papeleta por parte del Colegio Electoral Remoto.

Cerraría este proceso la inclusión del *hash* en el *Public Bulletin Board*, de manera que quedase contabilizado y se cumpliese la corrección y verificabilidad del proceso electoral. El elector puede comprobar su *hash* sin ser identificado por ninguna de las partes.

4. Modelado en Maude-NPA.

La herramienta por utilizar en este trabajo es Maude-NPA de Escobar, Meadows y Meseguer [3], ha sido desarrollada para evaluar las propiedades criptográficas de todo tipo de protocolos de comunicación. Para la evaluación, se modela el protocolo y se definen las propiedades en este; y se definen los ataques a los que enfrentar el protocolo.

Maude-NPA funciona construyendo las soluciones mediante búsquedas inversas que parten desde un estado final planteado. En esa búsqueda se generan masivamente todas las posibilidades, y Maude-NPA debe ir reduciendo los casos, eliminando las que no aportan en la solución.

Como se observará más adelante, cómo se implementan las distintas partes de la definición del protocolo tiene un impacto importante en el coste de las soluciones.

4.1. Sintaxis del protocolo

En Maude-NPA la codificación de los protocolos se hace en un único fichero con extensión ".maude" que se organiza en tres módulos distintos.

Cada uno de los módulos está dedicado a albergar una parte específica del modelado, simplificando y formalizando el esfuerzo. El comienzo de cada módulo se declara con el uso de una cadena reservada específica terminada en un símbolo "."; el fin de todos los módulos se indica con la cadena *endfm*.

El primer módulo de un fichero ".maude" está identificado por la cabecera:

fmod PROTOCOL-EXAMPLE-SYMBOLS is

protecting DEFINITION-PROTOCOL-RULES .

El módulo alberga la sintaxis para definir el protocolo, los tipos y los operadores.

El segundo módulo encapsula las propiedades criptográficas de los operadores a partir de *la cadena:*

fmod PROTOCOL-EXAMPLE-ALGEBRAIC is

protecting PROTOCOL-EXAMPLE-SYMBOLS .

El último módulo, etiquetado con:

fmod PROTOCOL-SPECIFICATION is

protecting PROTOCOL-EXAMPLE-SYMBOLS .

protecting DEFINITION-PROTOCOL-RULES .

protecting DEFINITION-CONSTRAINTS-INPUT .

es dónde se recogen el comportamiento del protocolo, las habilidades del atacante y la definición de cada ataque que se quiera plantear.

El tipo de datos original en Maude-NPA es el *Sort*, utilizado para representar los distintos tipos de datos necesarios en todo protocolo. Por supuesto, existen posteriores refinamientos de este tipo: los tipos *Msg*, *Fresh* y *Public*. Los usuarios utilizan estos subtipos y derivados de ellos.

Los usuarios no hacen uso del tipo *Sort* directamente; siempre se utilizarán los tipos -y subtipos derivados de ellos- de datos *Msg*, *Fresh* y *Public*. Estos han sido derivados de *Sort* para especificar tipos más complejos. Cada uno de los anteriores son tipos especiales y se han considerado con la siguiente mentalidad:

- *Msg*: Utilizado para representar los mensajes empleados en el protocolo. Si el protocolo en cuestión no hace más distinciones, no hará falta derivar desde él y solo se emplearán *Msg*. Si por el contrario conviene redefinir las distinciones, se podrá definir a partir de este otros *subsorts*. No puede ser un tipo vacío; siempre ha de definirse al menos un término de este tipo

  El uso de *subsorts* hace las búsquedas simbólicas mucho más eficientes. Esto se debe a que evita que Maude-NPA unifique términos incompatibles entre sí. En la mayoría de las ocasiones este es el efecto deseado; en otras, sin embargo, se puede desear generar ataques de confusión, en cuyo caso se evitará refinar en distintos *subsorts*.

- *Fresh*: Se emplea para representar datos cuyo valor ha de ser único a lo largo de la ejecución. Es el caso de claves, *nonces*, identificadores de sesión, etc… Solo las variables pueden adoptar este tipo.

- *Public*: Se usa para tipificar términos que se encontrarán públicamente disponibles a lo largo del protocolo. Por lo anterior es también conocido por el propio atacante. Es un subtipo de *Msg.* Ha de crearse al menos una instancia de este tipo.

Tras los tipos, el primer módulo contiene los operadores. Existen tres tipos de operadores: los prefijos, los infijos y los mix-fijos.

La sintaxis de los operadores se expresa con guiones bajos y la cadena identificadora del operador. Así los prefijos se reconocen por el operador situado antes del término – e.g., f, g, h… Los infijos sitúan sus operadores entre los dos términos -e.g., «_ + _». Cuando un usuario requiere expresar un operador más a medida, puede utilizar los *mix-fix*; con ellos pueden expresarse por ejemplo estructuras condicionales -e.g., *if_then_else_endif*.

En cada línea correspondiente a un operador binario pueden incluirse las propiedades que ha de satisfacer. Son los denominados atributos ecuacionales y expresan:

- Asociatividad: assoc
- Conmutatividad: comm
- Identidad indicada: id: keyword
- Asociatividad explícita, por la derecha gather(e E) o por la izquierda gather(E e) respectivamente.

El atributo debe incluirse en todas las declaraciones de operadores -excepto macros y constantes- para protegerlos ante reescrituras por parte de Maude-NPA; para ello se usa la palabra reservada *frozen*. En posteriores versiones de Maude-NPA está previsto que se considere de forma automática.

Se continúa el módulo con la definición de los *principal names*, es decir, los actores y piezas de información que participan en el protocolo. Es importante expresar correctamente sus roles y limitaciones, de lo contrario más adelante Maude-NPA no permitiría definir el protocolo. Por ejemplo, en ocasiones será necesario representar al atacante hablando directamente con alguno de los participantes; en ese caso hay que incluir al atacante entre los actores a definir.

En el caso de que el protocolo haga uso de elementos únicos -e.g., un *nonce*, clave, o en nuestro caso un voto o una máscara- será necesario el uso del modificador *Fresh* para asegurar esa unicidad.

4.2. Propiedades criptográficas

Aunque en este trabajo no vaya a hacerse uso de esta capacidad, Maude-NPA permite describir las propiedades criptográficas del protocolo. Solo se indicará que se pueden incluir tres tipos de propiedades algebráicas:

- Atributos ecuacionales del operador, o axiomas.
- Ecuaciones.
- Ecuaciones de metadatos

Todos ellos, en caso de necesitarse, han de incluirse en el segundo módulo.

Un caso típico de empleo explicado en Palop [20] o en el propio manual de Maude-NPA es la especificación de la relación existente -cancelación- entre las operaciones con las claves privada y pública de una cifra asimétrica. En ellas el empleo alternativo de las claves con la operación genera el texto en claro.

```
eq pk(A:Name,sk(A:Name,Z:Msg)) = Z:Msg [variant] .
eq sk(A:Name,pk(A:Name,Z:Msg)) = Z:Msg [variant] .
```

Ilustración 5. Ejemplo de propiedad criptográfica de cancelación.

4.3. Especificación de strands

El tercero de los módulos recoge, en lo que se denomina *strands,* las capacidades del intruso, la propia definición del protocolo y por último los ataques a probar.

Un *strand* es una secuencia de mensajes enviados o recibidos[9] por uno de los actores principales -incluido el atacante- en la ejecución del protocolo con el siguiente formato; donde el carácter «+» indica un mensaje enviado y «-» un mensaje recibido.

$$:: r_1 , \dots , r_j :: [m_1^{\pm}, \dots , m_i^{\pm} \mid m_{i+1}^{\pm}, \dots , m_k^{\pm}]$$

Ilustración 6. Generalización del concepto Strand.

La barra «|» es empleada para diferenciar entre el pasado, presente y el futuro cuando el *strand* aparece en una descripción de estado. Todo lo que queda a la izquierda de «|» ha ocurrido en el pasado, mientras que, si se encuentra a la derecha, habrá de ocurrir en el futuro.

Al definir la especificación del protocolo se asume que la barra se encuentra al principio de la cadena, y a su izquierda se indica *nil* pues no ha habido nada aún.

Así, por ejemplo, el caso particular del protocolo NSPK explicado en [3] se especifica usando:

```
:: r ::
[ nil | +(pk(B,A ; n(A,r))), -(pk(A,n(A,r) ; N)), +(pk(B, N)), nil ]
:: r ::
[ nil | -(pk(B,A ; N)), +(pk(A, N ; n(B,r))), -(pk(B,n(B,r))), nil ]
```

Ilustración 7. Especificación con strands NSPK

Que de manera informal puede exponerse como:

1. $A \rightarrow B : pk(B, A; N_A)$
2. $B \rightarrow A: pk(A, N_A; N_B)$
3. $A \rightarrow B : pk(B, N_B)$

Ilustración 8. Especificación matemática informal NSPK.

[9] Denotándose con «+» o «–» respectivamente.

O gráficamente como:

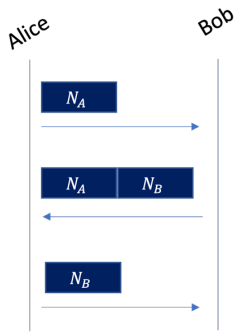

Ilustración 9. NSPK para Alice y Bob.

Por otra parte, las capacidades del intruso se especifican siguiendo el modelo de reglas Dolev-Yao [23]; en él siempre hay una secuencia de mensajes recibidos, seguidos de un mensaje enviado. De esta manera lo que se está expresando son los pre-requisitos para que el atacante debe encontrar para que pueda realizar una acción.

El modelo de atacante en Maude-NPA se define como un *man-in-the-middle*, que puede observar lo que ocurre en el medio de comunicación, en ambas direcciones. Esto le hace recibir también la denominación de intruso.

Una buena definición de las capacidades del atacante puede reducir el espacio de búsqueda, ayudando a acelerar la obtención de resultados. Maude-NPA proporciona libertad a la hora de definir estas capacidades.

```
eq STRANDS-DOLEVYAO
= :: nil :: [ nil | -(X), -(Y), +(X ; Y), nil ] &
:: nil :: [ nil | -(X ; Y), +(X), nil ] &
:: nil :: [ nil | -(X ; Y), +(Y), nil ] &
:: nil :: [ nil | -(X), +(sk(i,X)), nil ] &
:: nil :: [ nil | -(X), +(pk(Ke,X)), nil ] &
:: nil :: [ nil | +(A), nil ]
[nonexec] .
```

Ilustración 10. Ejemplo extraído de Palop [20] de especificación siguiendo Dolev-Yao.

En la ilustración anterior se observa cómo entre la cabecera y el cierre lo que se expresa es que; el atacante puede componer y descomponer mensajes en partes; aplicar el operador de cifra con clave pública, o privada; e introducir mensajes propios en el canal.

4.4. Realización de ataques

Finalmente, el módulo recoge la especificación de los ataques que se desee plantear. Estos ataques, que reciben el nombre de *attack-states*, esbozan la situación final deseada; Maude-NPA se encargará de generar un árbol con todos los estados posibles y encontrar el camino válido hasta un estado inicial.

Es posible definir distintos ataques, identificando a cada uno con un código numérico. Cada patrón contiene distintas secciones delimitadas por «||» [3].

```
eq ATTACK-STATE(id_numérico) =
   Strands de desarrollo del protocolo
|| Conocimiento del atacante
|| Secuencia de mensajes
|| Datos adicionales
|| Never Patterns
[nonexec].
```

Ilustración 11. Esquema de attack-state tipo.

Aunque solo se da contenido a las dos primeras; estas corresponden al conjunto de *strands* que se espera que sucedan para ese *attack-state* y al conocimiento esperado del atacante. Las otras secciones, aunque tienen funcionalidad interna, deben tener el símbolo *nil*.

La última de las secciones recibe el nombre de *never pattern*; tiene una consideración especial y se utiliza para especificar situaciones que nunca han de producirse. Es posible incluir un segundo *never pattern* que restringa aún más al primero. Cada *nevern pattern* es considerado por Maude-NPA independientemente.

Los *never pattern* pueden usarse para constreñir el espacio de búsqueda, sin embargo, al usarlos hay que ser muy cuidadosos; puesto que se pueden acabar considerando los ataques como falsos negativos debido a un exceso de poda.

5. Codificación del protocolo TAVS en Maude-NPA

Al evaluar cualquier objeto -también protocolos- hay que considerar qué grado de precisión es necesario para conseguir representarlo adecuadamente. Existe el riesgo de sobrerrepresentar; añadiendo características que realmente no van a aportar valor al modelo y van a suponer costes adicionales. Hay que recordar que el objetivo es la obtención de resultados dentro de un margen de tiempo razonable, no los cálculos per se.

Aunque el protocolo TAVS está diseñado de manera elegantemente sencilla, al estudiarlo se observan características que no aportan valor para este estudio. En concreto se ha prescindido de representar:

- Los flujos que componen los mensajes de control, debido a que no transportan información de valor para los casos del atacante.
- Las acciones de publicación en el *Revoked Board* y el *Public Bulletin Board*, ya que no hay manera de ligarlas a la obtención de conocimiento valioso o beneficio bajo ninguno de los ataques a elaborar.

A pesar de estas consideraciones, y por lo referido, el modelo adoptado no descarta gran cosa, y el resultado no se ve afectado negativamente. Las interacciones entre actores relevantes siguen representadas adecuadamente.

El cifrado RSA que se asume en Larriba, Sempere y López [2] es representado de manera genérica, ya que en realidad del protocolo se ha especificado para no tener dependencias específicas.

Se ha respetado el esquema de cifra asimétrica con su par de claves para la IA. En la evolución del ataque de confidencialidad se ha representado el cifrado del canal con un esquema de cifra asimétrico para todos los participantes.

5.1. Símbolos y propiedades

Tal y como se ha presentado en los apartados previos del trabajo, lo primero que se ha de especificar son los sorts, subsorts, roles principales y operadores.

Antes de iniciar la explicación hay que aclarar que en algunos casos puede encontrarse una cadena con dos variaciones: inicio con mayúscula o todo el minúscula (e.g., Hash y hash). Cuando esto ocurre, el término con mayúscula hace referencia a un campo o valor; por el contrario, el término en minúsculas siempre hará referencia a la función. Es decir, uno es un valor y el otro un operador.

```
fmod PROTOCOL-EXAMPLE-SYMBOLS is
  protecting DEFINITION-PROTOCOL-RULES .

  --- Información Sort
  sorts Name Key Mask Choice Hash Elector IdenAuth RemotePS Intruder .
  subsort Name Key Mask Choice Hash < Msg .
  subsort Elector IdenAuth RemotePS Intruder < Name .
  subsort Name < Public .
  subsort Name < Key .

--- Roles
  op Elec : -> Elector .
  op Auth : -> IdenAuth .
  op Rps : -> RemotePS .
  op intruder : -> Intruder .

--- Operadores
  op choice : Name Fresh -> Choice .
  op hash : Msg -> Hash .
  op mask : Name Fresh -> Mask .

--- Operadores de cifrado público y privado
  op pk : Key Msg -> Msg [frozen] .
  op sk : Key Msg -> Msg [frozen] .

--- Operador concatenación
  op _;_ : Msg Msg -> Msg [gather (e E) frozen] .

--- Operador multiplicación exponencial
  op _*_ : Msg Msg -> Msg [frozen] .

endfm
```

Ilustración 12. Módulo 1: Sorts, subsorts y operadores. Roles principales.

Además del uso de tipos por defecto como Msg, hay que refinar los subtipos de datos que dan soporte a las piezas de información relevantes en el protocolo:

- *Choice*: codifica el sentido de voto del elector. Solo conocido por él. Es un valor único, por lo que debe incorporarse como *Fresh*.
- *Mask*: se utiliza para ocultar temporalmente el sentido de voto a la autoridad de identificación. Solo conocido por el elector y el *Colegio Electoral Remoto*. Es también un valor único, por lo que ha de incluirse como *Fresh*.
- *Hash*: es el valor calculado con la función *hash* a partir del *choice* y el *mask*.

Se definen los siguientes roles de participantes, elector, Autoridad de Identificación y Colegio Electoral Remoto. Adviértase como ya no aparecen ninguno de los dos *boards* que sí están en el protocolo original: *Revoked Board* y *Public Bulletin Board*. Asímismo se incorpora al atacante.

Hay una doble tipificación de los roles, como *Name* y como *key*. Esto se hace para poder utilizarlos tanto al especificar origen o destino de un mensaje, como quien realiza un cifrado. Esto simplifica el código y permite evitar fallos de codificación y confusiones, pues con ello se limitan el número de opciones a usar.

Han de declararse las operaciones específicas que se usan en el protocolo. En este caso el cifrado asimétrico, la concatenación y la multiplicación exponencial.

La propiedad homomórfica respecto de la cifra asimétrica está representada implícitamente en Maude-NPA. No hay propiedades algebraicas específicas que considerar. Por ello el segundo módulo es reducido y solo alberga la definición de algunas variables.

```
fmod PROTOCOL-EXAMPLE-ALGEBRAIC is
  protecting PROTOCOL-EXAMPLE-SYMBOLS .

var Z : Msg .
var Ke : Key .

endfm
```

Ilustración 13. Módulo 2: Propiedades algebráicas.

5.2. Capacidades del atacante

Como ya se avanzó, las habilidades del atacante se codifican siguiendo el modelo de Dolev-Yao [23]. Dicha codificación corresponde a la segunda de las partes que componen el tercer bloque, tras la definición de variables.

```
fmod PROTOCOL-SPECIFICATION is
  protecting PROTOCOL-EXAMPLE-SYMBOLS .
  protecting DEFINITION-PROTOCOL-RULES .
  protecting DEFINITION-CONSTRAINTS-INPUT .

  --- Clave genérica
  var K : Key .

  --- Mensajes genéricos
  vars M M1 M2 : Msg .
  var Msk : Mask .
  vars Choice Mask : Fresh .
  var A : Name .
  var CH : Choice .

--- Roles genéricos para los STRANDS-PROTOCOL
  vars ELEC AUTH RPS : Name .
```

Ilustración 14. Tercer modulo: declaración de inicio y variables.

En concreto, se ha caracterizado al atacante sobre el protocolo TAVS con las capacidades de concatenar y separar mensajes; realizar multiplicaciones exponenciales; si conoce la máscara -o implícitamente su inversa- desenmascarar mensajes; cifrar y descifrar genéricamente mensajes si recibe una clave para usar; descifrar mensajes cuya clave este públicamente disponible; utilizar la función *hash* asociada al protocolo; enviar sus propios mensajes.

Hay dos *strands* distintivos y particularmente interesantes – los últimos. El primero de ellos indica que el atacante es capaz de generar sus propios *choice*, asociados a su identidad. El segundo que puede generar sus propias máscaras. Aunque no serán usadas aquí, estas dos reglas podrán ser de utilidad en trabajos posteriores.

```
eq STRANDS-DOLEVYAO

 = :: nil :: [ nil | -(M1 ; M2), +(M1), nil ] &
   :: nil :: [ nil | -(M1 ; M2), +(M2), nil ] &
   :: nil :: [ nil | -(M1), -(M2), +(M1 ; M2), nil ] &
   :: nil :: [ nil | -(M1), -(M2), +(M1 * M2), nil ] &
   :: nil :: [ nil | -(M * Msk), -(Msk), +(M), nil ] & --- Only for known masks
   :: nil :: [ nil | -(K), -(M), +(pk(K,M)), nil ] &
   :: nil :: [ nil | -(K), -(M), +(sk(K,M)), nil ] &
   :: nil :: [ nil | -(sk(K,M)), +(M), nil ] & --- Possible because public key
   :: nil :: [ nil | -(M), +(hash(M)), nil ] &
   :: nil :: [ nil | +(A), nil ] &
   :: Choice :: [ nil | +(choice(intruder,Choice)), nil ] &
   :: Mask :: [ nil | +(mask(intruder,Mask)), nil ]

[nonexec] .
```

Ilustración 15. Tercer modulo: reglas Dolev-Yao de capacidades del atacante.

El resto de strands (comenzando por el principio) se modelan las siguientes capacidades:

- Extraer la parte derecha de una cadena de mensaje.
- Extraer la parte izquierda de una cadena de mensaje.
- Componer un mensaje a partir de dos originales.
- Realizar la operación multiplicación exponencial de dos mensajes.
- Aplicar la operación inversa de enmascarado si se conoce la máscara.
- Aplicar una clave pública conocida a la firma o cifra de un mensaje.
- Aplicar una clave privada conocida a la firma o cifra de un mensaje.
- Obtener mensajes cifrados con clave secreta cuya clave pública sea conocida.
- Obtener el hash de un mensaje.
- Introducir cadenas propias en el medio de comunicación.

La definición de los strands Dolev-Yao tienen un impacto realmente significativo sobre el coste para encontrar la solución. La filosofía de Maude-NPA hace que se genere todo el espacio de búsqueda posible a partir de ellas de forma sistemática. Así, por ejemplo, la existencia de las tres primeras reglas representadas aquí hace que se generen un enorme número de posibilidades de concatenación y división de mensajes, que posteriormente han de ser evaluados.

5.3. Strands de TAVS

Una vez planteadas las capacidades del atacante, es necesario detallar el comportamiento propio del protocolo TAVS. Para ello se emplea la siguiente serie de *strands*.

```
eq STRANDS-PROTOCOL
 = :: Choice, Mask :: *** STRANDS DEL ELECTOR ***
 [ nil |
       +(ELEC ; AUTH ; ((choice(ELEC,Choice) ; hash(choice(ELEC,Choice) ; mask(ELEC,Mask))) * pk(AUTH,mask(ELEC,Mask)))),
       -(AUTH ; ELEC ; (sk(AUTH,(choice(ELEC,Choice) ; hash(choice(ELEC,Choice) ; mask(ELEC,Mask)))) * mask(ELEC,Mask))),
       +(ELEC ; RPS ; (sk(AUTH,(choice(ELEC,Choice) ; hash(choice(ELEC,Choice) ; mask(ELEC,Mask)))) * mask(ELEC,Mask)) ; mask(ELEC,Mask)),
  nil ] &

 :: nil :: *** STRANDS DEL IDENTIFICATION AUTHORITY ***
 [ nil |
       -(ELEC ; AUTH ; ((CH ; hash(CH ; Msk)) * pk(AUTH,Msk))),
       +(AUTH ; ELEC ; (sk(AUTH,(CH ; hash(CH ; Msk))) * Msk)),
 nil ] &
 :: nil :: *** STRANDS DEL REMOTE POLLING STATION ***
 [ nil |
       -(ELEC ; RPS ; (sk(AUTH,(CH ; hash(CH ; Msk))) * Msk) ; Msk),
 nil ]
[nonexec] .
```

Ilustración 16. Módulo 3: Bloque de strands de definición del protocolo TAVS.

Si se hace una representación gráfica de los *strands*, se encontrarán una serie de esquemas, que forman un conjunto muy similar a la ilustración 1; esta vez mucho más rica en los detalles de los mensajes. A su vez, se han eliminado las partes del protocolo que no aportan interés, por lo que se aprecia también más limpia.

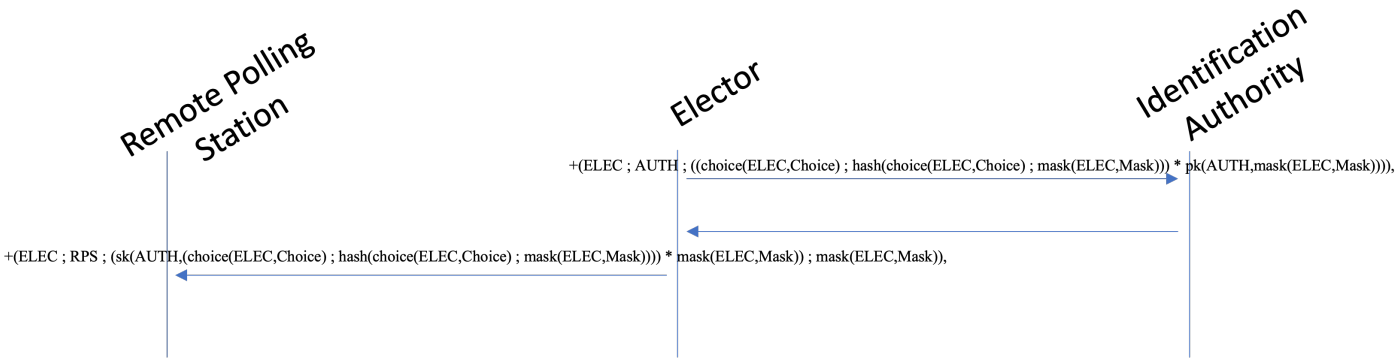

Ilustración 17. Vista de los strands de TAVS desde el Elector.

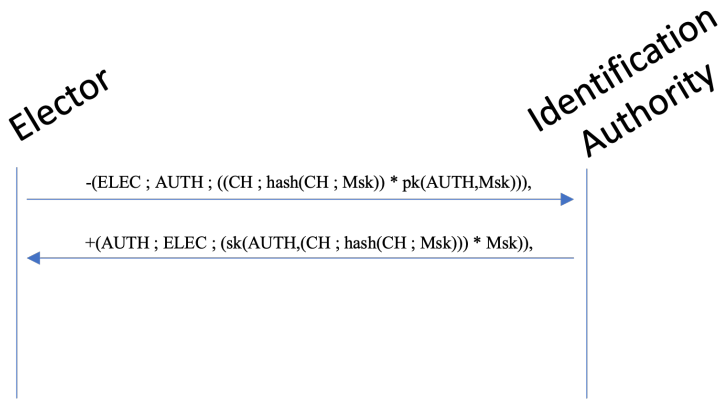

Ilustración 18. Vista de los strands de TAVS desde la Identity Authority.

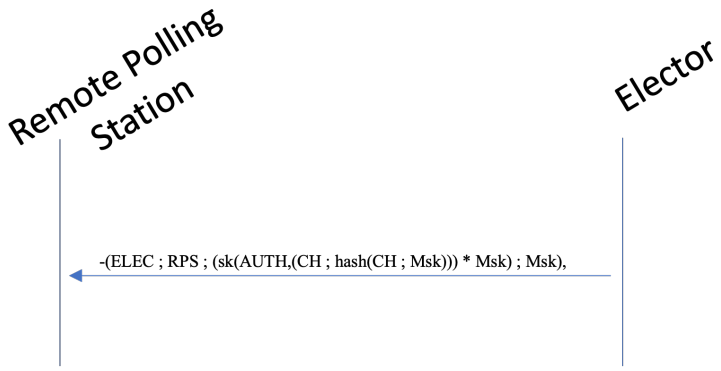

Ilustración 19. Vista de los strands ed TAVS desde la Colegio Electoral Remoto.

Se puede observar cómo los distintos participantes definen únicamente los strands que afectan a su interacción.

Todo el modelo corresponde a una votación concreta y única por parte de un elector -recordemos la categorización de *Choice* y *Mask* como *Fresh*.

6. Análisis del protocolo

Este capítulo recoge el proceso de ejecución del modelado. Los objetivos del trabajo se plasmarán en cada uno de los siguientes apartados.

Se han planteado cinco escenarios que cubren:

- La comprobación funcional del modelo
- La confidencialidad sin protección del medio de comunicación
- La confidencialidad con protección del medio de comunicación
- La autenticación del elector
- La autenticación de la autoridad.

Los ataques sobre la autenticación se han realizado asumiendo el modelo de desprotección, pues una vez comprobada la robustez del modelo protegido, se asume que este se extiende a cualquier variante que quiera implementar.

6.1. Ejecución regular

Es habitual implementar inicialmente un *attack pattern* que en realidad no corresponda a una intención de ataque; si no más bien a la voluntad de comprobar que el protocolo ha sido correctamente codificado. Sirve como comprobación, permite la compilación, detecta los errores y asegura la ejecución esperada.

```
eq ATTACK-STATE(0) *** EJECUCIÓN STANDARD ***

  = :: Choice, Mask :: *** EJECUCIÓN DEL ELEC ***

    [ nil,

+(Elec ; Auth ; ((choice(Elec,Choice) ; hash(choice(Elec,Choice) ; mask(Elec,Mask))) * pk(Auth,mask(Elec,Mask)))),
-(Auth ; Elec ; (sk(Auth,(choice(Elec,Choice) ; hash(choice(Elec,Choice) ; mask(Elec,Mask)))) * mask(Elec,Mask))),
+(Elec ; Rps ; (sk(Auth,(choice(Elec,Choice) ; hash(choice(Elec,Choice) ; mask(Elec,Mask)))) * mask(Elec,Mask)) ;
mask(Elec,Mask))
    | nil ] &
    :: nil :: *** EJECUCIÓN DEL IDENTIFICATION AUTHORITY ***
   [ nil,
-(Elec ; Auth ; ((choice(Elec,Choice) ; hash(choice(Elec,Choice) ; mask(Elec,Mask))) * pk(Auth,mask(Elec,Mask)))),
+(Auth ; Elec ; (sk(Auth,(choice(Elec,Choice) ; hash(choice(Elec,Choice) ; mask(Elec,Mask)))) * mask(Elec,Mask)))
   | nil ] &
   :: nil :: *** EJECUCIÓN DEL REMOTE POLLING STATION ***
   [ nil,
-(Elec ; Rps ; (sk(Auth,(choice(Elec,Choice) ; hash(choice(Elec,Choice) ; mask(Elec,Mask)))) * mask(Elec,Mask)) ;
mask(Elec,Mask))
    | nil ]
    || empty
    || nil
    || nil
    || nil

 [nonexec] .
```

Ilustración 20. Código correspondiente a la ejecución «standard» de TAVS.

En nuestro caso Maude-NPA solo necesitó 4 niveles de profundidad para obtener la solución. Dado que no se le ha solicitado a Maude-NPA ningún estado final especial, la ejecución consiste en desarrollar el protocolo.

```
Maude> red summary(0,0) .
reduce in MAUDE-NPA : summary(0, 0) .
rewrites: 89448476 in 122933ms cpu (123745ms real) (727616 rewrites/second)
result Summary: States>> 1 Solutions>> 0
Maude> red summary(0,1) .
reduce in MAUDE-NPA : summary(0, 1) .
rewrites: 2673372 in 3405ms cpu (3435ms real) (784950 rewrites/second)
result Summary: States>> 3 Solutions>> 0
Maude> red summary(0,2) .
reduce in MAUDE-NPA : summary(0, 2) .
rewrites: 3829634 in 5721ms cpu (6035ms real) (669287 rewrites/second)
result Summary: States>> 6 Solutions>> 0
Maude> red summary(0,3) .
reduce in MAUDE-NPA : summary(0, 3) .
rewrites: 9134839 in 10455ms cpu (10603ms real) (873719 rewrites/second)
result Summary: States>> 11 Solutions>> 1
Maude>
```

Ilustración 21. Ejecución del caso «standard» hasta encontrar solución.

Durante la ejecución, cada uno de los niveles generó cierto número de estados intermedios: 1, 3, 6, 11. En el cuarto nivel de ejecución, Maude-NPA ya fue capaz de considerar la solución -en realidad llegar hasta el final de la ejecución estándar.

```
Maude> red initials (0,3) .
reduce in MAUDE-NPA : initials(0,3) .
rewrites: 476 in 0ms cpu (0ms real) (~ rewrites/second)
result ShortIdSystem: < 1 . 2 . 1 . 1 > (
:: nil ::
[ nil |
   -(Elec ; Auth ; ((choice(Elec, #0:Fresh) ; hash(choice(Elec, #0:Fresh) ; mask(Elec, #1:Fresh))) * pk(Auth, mask(Elec, #1:Fresh)))),
   +(Auth ; Elec ; (sk(Auth, choice(Elec, #0:Fresh) ; hash(choice(Elec, #0:Fresh) ; mask(Elec, #1:Fresh))) * mask(Elec, #1:Fresh))), nil]  &
:: nil ::
[ nil |
   -(Elec ; Rps ; (sk(Auth, choice(Elec, #0:Fresh) ; hash(choice(Elec, #0:Fresh) ; mask(Elec, #1:Fresh))) * mask(Elec, #1:Fresh)) ; mask(Elec, #1:Fresh)), nil]  &
:: #1:Fresh,#0:Fresh ::
[ nil |
   +(Elec ; Auth ; ((choice(Elec, #0:Fresh) ; hash(choice(Elec, #0:Fresh) ; mask(Elec, #1:Fresh))) * pk(Auth, mask(Elec, #1:Fresh)))),
   -(Auth ; Elec ; (sk(Auth, choice(Elec, #0:Fresh) ; hash(choice(Elec, #0:Fresh) ; mask(Elec, #1:Fresh))) * mask(Elec, #1:Fresh))),
   +(Elec ; Rps ; (sk(Auth, choice(Elec, #0:Fresh) ; hash(choice(Elec, #0:Fresh) ; mask(Elec, #1:Fresh))) * mask(Elec, #1:Fresh)) ; mask(Elec, #1:Fresh)), nil] )
|
(Elec ; Auth ; ((choice(Elec, #0:Fresh) ; hash(choice(Elec, #0:Fresh) ; mask(Elec, #1:Fresh))) * pk(Auth, mask(Elec, #1:Fresh)))) !inI,
(Elec ; Rps ; (sk(Auth, choice(Elec, #0:Fresh) ; hash(choice(Elec, #0:Fresh) ; mask(Elec, #1:Fresh))) * mask(Elec, #1:Fresh)) ; mask(Elec, #1:Fresh)) !inI,
(Auth ; Elec ; (sk(Auth, choice(Elec, #0:Fresh) ; hash(choice(Elec, #0:Fresh) ; mask(Elec, #1:Fresh))) * mask(Elec, #1:Fresh))) !inI
|
+(Elec ; Auth ; ((choice(Elec, #0:Fresh) ; hash(choice(Elec, #0:Fresh) ; mask(Elec, #1:Fresh))) * pk(Auth, mask(Elec, #1:Fresh)))),
-(Elec ; Auth ; ((choice(Elec, #0:Fresh) ; hash(choice(Elec, #0:Fresh) ; mask(Elec, #1:Fresh))) * pk(Auth, mask(Elec, #1:Fresh)))),
+(Auth ; Elec ; (sk(Auth, choice(Elec, #0:Fresh) ; hash(choice(Elec, #0:Fresh) ; mask(Elec, #1:Fresh))) * mask(Elec, #1:Fresh))),
-(Auth ; Elec ; (sk(Auth, choice(Elec, #0:Fresh) ; hash(choice(Elec, #0:Fresh) ; mask(Elec, #1:Fresh))) * mask(Elec, #1:Fresh))),
+(Elec ; Rps ; (sk(Auth, choice(Elec, #0:Fresh) ; hash(choice(Elec, #0:Fresh) ; mask(Elec, #1:Fresh))) * mask(Elec, #1:Fresh)) ; mask(Elec, #1:Fresh)),
-(Elec ; Rps ; (sk(Auth, choice(Elec, #0:Fresh) ; hash(choice(Elec, #0:Fresh) ; mask(Elec, #1:Fresh))) * mask(Elec, #1:Fresh)) ; mask(Elec, #1:Fresh))
|
nil
Maude>
```

Ilustración 22. Resultados tras la ejecución «standard».

Es posible solicitar a Maude-NPA la traza de mensajería que ocurre desde el estado inicial al estado final. Esta herramienta es de especial utilidad al detallar el camino que un ataque sigue hasta obtener el resultado que busca. Su interpretación proporciona la ejecución detallada del ataque paso a paso.

En este caso el *attack-pattern* como ya se ha indicado corresponde a la ejecución estándar. En la imagen superior, correspondiente a la captura del resultado se aprecian todos los mensajes desde el punto de vista de los tres roles. Se observa que el protocolo para cada uno de los participantes se ha ejecutado como corresponde, sin errores ni anomalías.

6.2. Confidencialidad

Para cualquier protocolo criptográfico, o en este caso de voto electrónico es vital comprobar la confidencialidad. Esta propiedad sirve de puntal para proporcionar otras como la privacidad o el repudio[10].

TAVS se centra en la funcionalidad y requisitos esperados y necesarios a un sistema de voto electrónico. Todo aquello necesario, pero no suficiente. El resto se plantea como requisitos al sistema, implementables de una u otra forma, pero que no aportan un hecho diferencial a TAVS. Un ejemplo de esta concepción es que se considera el medio de comunicación seguro, pero no aborda cómo.

Al plantear el primer *attack-pattern* del protocolo TAVS, se decidió comprobar el impacto que pueden llegar a suponer los prerrequisitos en la validez del modelo. Por eso se han desarrollaron dos experimentos, en el primero -este punto 6.1.2- se evaluó la confidencialidad sin considerar el medio seguro. En el segundo se implementó la protección como una capa más en torno al protocolo.

```
eq ATTACK-STATE(3) ***CONFIDENCIALIDAD SIN PROTECCION***
    = :: Choice, Mask :: *** EJECUCIÓN DEL ELEC ***
    [ nil,
+(Elec ; Auth ; ((choice(Elec,Choice) ; hash(choice(Elec,Choice) ; mask(Elec,Mask))) * pk(Auth,mask(Elec,Mask)))),
-(Auth ; Elec ; (sk(Auth,(choice(Elec,Choice) ; hash(choice(Elec,Choice) ; mask(Elec,Mask)))) * mask(Elec,Mask))),
+(Elec ; Rps ; (sk(Auth,(choice(Elec,Choice) ; hash(choice(Elec,Choice) ; mask(Elec,Mask)))) * mask(Elec,Mask)) ; mask(Elec,Mask))
    | nil ]
   || choice(Elec,Choice) inI, empty
   || nil
   || nil
   || nil

 [nonexec] .
```

Ilustración 23. Attack-pattern sobre la confidencialidad -medio de comunicación no seguro- de TAVS

En concreto, para este *attack-pattern* se plantea el paso de mensajes entre los participantes desde el punto de vista del elector. Esto es así porqué el elector ocupa una posición central privilegiada, desde la que transmite la información que crea él, y retransmite la que producen los demás. Por ello los otros dos roles son implícitos en este patrón.

Se añade al *attack-pattern en su sección dos* la condición a buscar: que el intruso sea capaz de conocer o calcular la elección -choice- del elector.

Se obtuvo resultado positivo. Sí, existe un camino -si el medio de comunicación no está protegido- en el que el atacante es capaz de recuperar el sentido de voto del elector. Maude-NPA necesitó de 13 niveles para localizar una solución al reto planteado.

[10] Aquí estaríamos hablando de la posibilidad de negar un voto concreto por parte del elector. No confundir con el «no repudio».

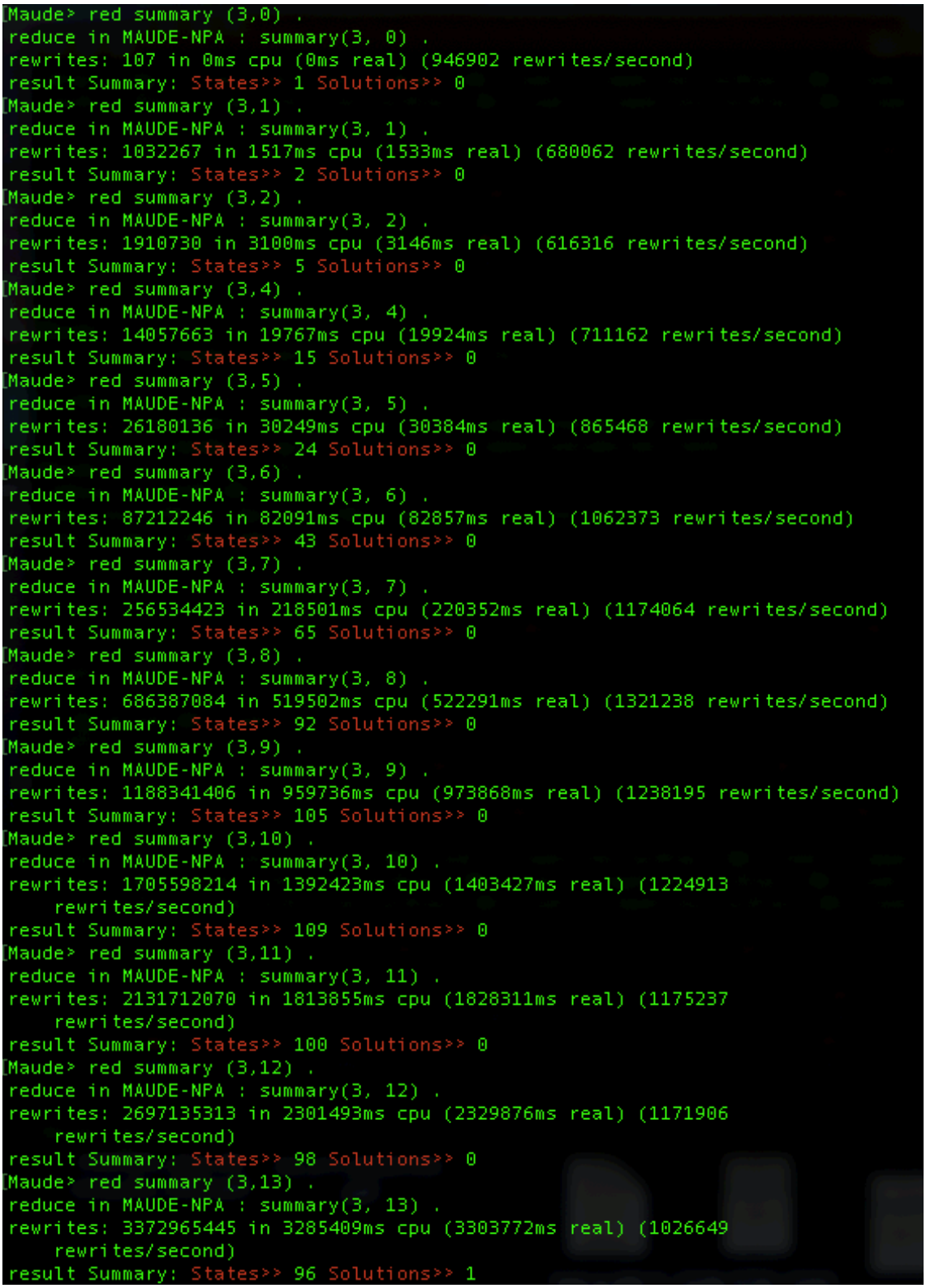

Ilustración 24. Cálculo del Attack-pattern sobre la confidencialidad -medio de comunicación no seguro.

Si se continua la ejecución a partir del decimotercer nivel se aprecia la optimización del número de estados necesario para la solución. Maude-NPA va podando los estados que no aportan valor.

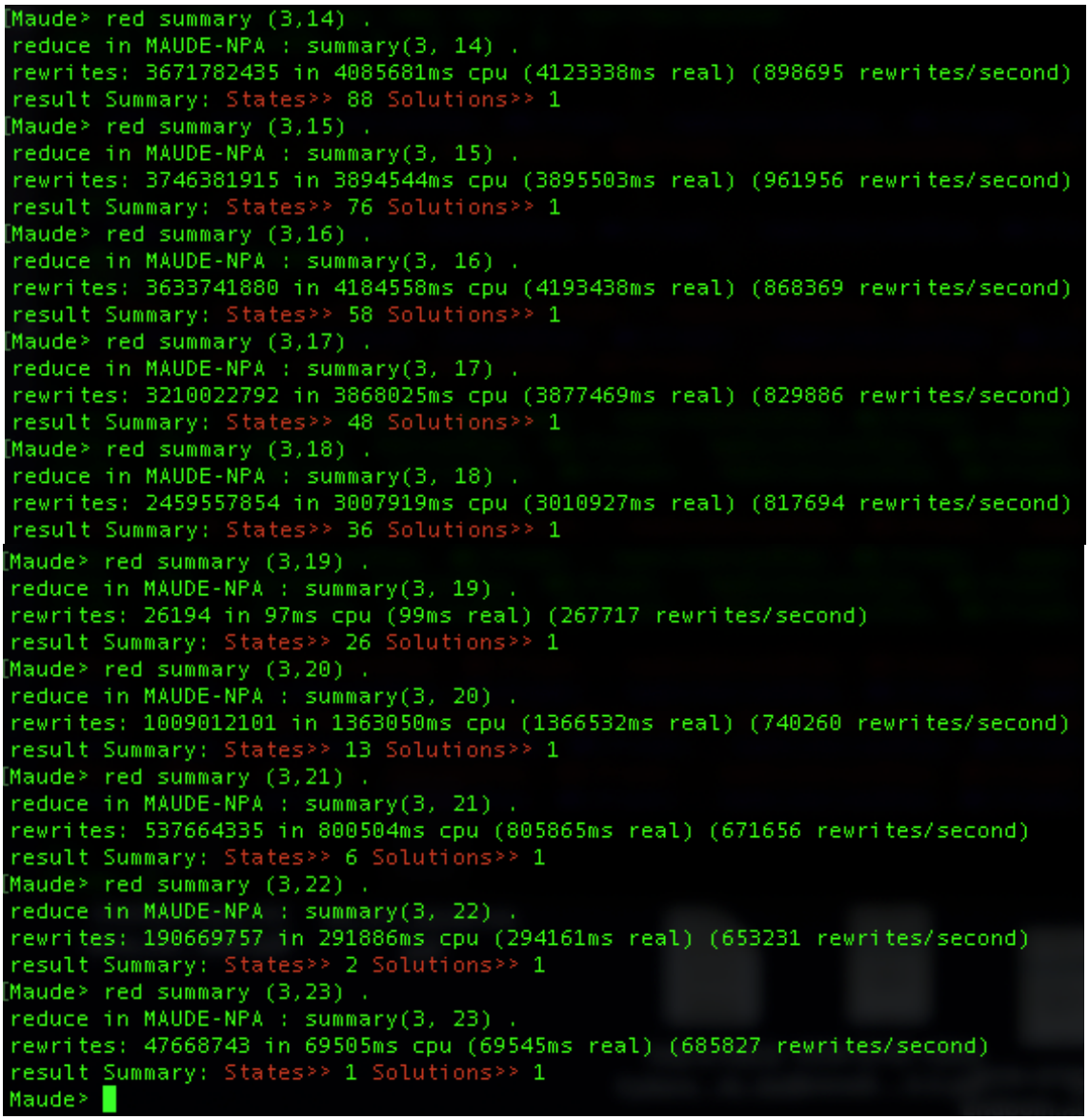

Ilustración 25. Optimización del resultado.

El interés ahora radica en conocer la mensajería y las acciones que el atacante ha llevado a cabo para obtener el *Choice*.

Para observar el desarrollo del ataque se utiliza el comando «*red initials (3,13) .*», que genera el siguiente resultado.

Ilustración 26. Ejecución del ataque sobre la confidencialidad en medio de comunicación no seguro.

Entre todo el contenido que presenta la herramienta, el interés se centra en la última parte. Donde se explicitan los pasos del protocolo ejecutados y los propios del atacante. El bloque dispone de los mensajes en sentido origen y destino -se envían o se reciben- coloreados y marcados «+» para envío y «-» para recepción para diferenciarlos.

```
+(Elec ; Auth ; ((choice(Elec, #0:Fresh) ; hash(choice(Elec, #0:Fresh) ; mask(Elec, #1:Fresh))) * pk(Auth, mask(Elec, #1:Fresh)))),
-(Elec ; Auth ; ((choice(Elec, #0:Fresh) ; hash(choice(Elec, #0:Fresh) ; mask(Elec, #1:Fresh))) * pk(Auth, mask(Elec, #1:Fresh)))),
+(Auth ; Elec ; (sk(Auth, choice(Elec, #0:Fresh) ; hash(choice(Elec, #0:Fresh) ; mask(Elec, #1:Fresh))) * mask(Elec, #1:Fresh))),
-(Auth ; Elec ; (sk(Auth, choice(Elec, #0:Fresh) ; hash(choice(Elec, #0:Fresh) ; mask(Elec, #1:Fresh))) * mask(Elec, #1:Fresh))),
+(Elec ; Rps ; (sk(Auth, choice(Elec, #0:Fresh) ; hash(choice(Elec, #0:Fresh) ; mask(Elec, #1:Fresh))) * mask(Elec, #1:Fresh)) ; mask(Elec, #1:Fresh)),
-(Elec ; Rps ; (sk(Auth, choice(Elec, #0:Fresh) ; hash(choice(Elec, #0:Fresh) ; mask(Elec, #1:Fresh))) * mask(Elec, #1:Fresh)) ; mask(Elec, #1:Fresh)),
+(Rps ; (sk(Auth, choice(Elec, #0:Fresh) ; hash(choice(Elec, #0:Fresh) ; mask(Elec, #1:Fresh))) * mask(Elec, #1:Fresh)) ; mask(Elec, #1:Fresh)),
-(Rps ; (sk(Auth, choice(Elec, #0:Fresh) ; hash(choice(Elec, #0:Fresh) ; mask(Elec, #1:Fresh))) * mask(Elec, #1:Fresh)) ; mask(Elec, #1:Fresh)),
+((sk(Auth, choice(Elec, #0:Fresh) ; hash(choice(Elec, #0:Fresh) ; mask(Elec, #1:Fresh))) * mask(Elec, #1:Fresh)) ; mask(Elec, #1:Fresh)),
-((sk(Auth, choice(Elec, #0:Fresh) ; hash(choice(Elec, #0:Fresh) ; mask(Elec, #1:Fresh))) * mask(Elec, #1:Fresh)) ; mask(Elec, #1:Fresh)),
+(mask(Elec, #1:Fresh)),
-((sk(Auth, choice(Elec, #0:Fresh) ; hash(choice(Elec, #0:Fresh) ; mask(Elec, #1:Fresh))) * mask(Elec, #1:Fresh)) ; mask(Elec, #1:Fresh)),
+(sk(Auth, choice(Elec, #0:Fresh) ; hash(choice(Elec, #0:Fresh) ; mask(Elec, #1:Fresh))) * mask(Elec, #1:Fresh)),
resuscitated(mask(Elec, #1:Fresh)),
-(sk(Auth, choice(Elec, #0:Fresh) ; hash(choice(Elec, #0:Fresh) ; mask(Elec, #1:Fresh))) * mask(Elec, #1:Fresh)),
-(mask(Elec, #1:Fresh)),
+(sk(Auth, choice(Elec, #0:Fresh) ; hash(choice(Elec, #0:Fresh) ; mask(Elec, #1:Fresh)))),
-(sk(Auth, choice(Elec, #0:Fresh) ; hash(choice(Elec, #0:Fresh) ; mask(Elec, #1:Fresh)))),
+(choice(Elec, #0:Fresh) ; hash(choice(Elec, #0:Fresh) ; mask(Elec, #1:Fresh))),
-(choice(Elec, #0:Fresh) ; hash(choice(Elec, #0:Fresh) ; mask(Elec, #1:Fresh))),
+(choice(Elec, #0:Fresh))
```

Ilustración 27. Detalle del bloque de interés.

Las seis primeras líneas corresponden al paso de mensajes entre los roles legítimos, es la ejecución de TAVS. A partir de la línea siete se aprecia la actividad del intruso.

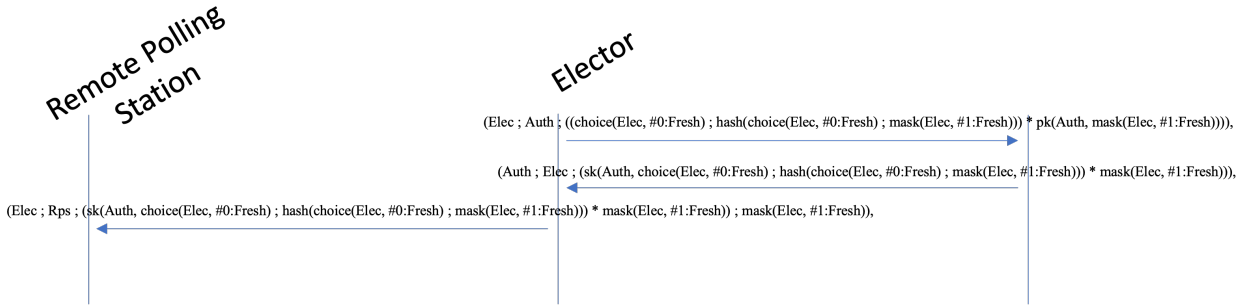

Ilustración 28. Participación legítima.

Inicialmente recibe o escucha una copia del mensaje[11] desde el Elector al RPS, y lo desensambla, extrayendo el destinatario.

Es interesante como separa, haciendo la operación de desconcatenación, la máscara en claro del grueso del mensaje. Guarda ambos -y suspende el progreso del estado asociado- hasta poder buscarle utilidad.

[11] Se han simplificado los mensajes para conseguir mayor claridad.

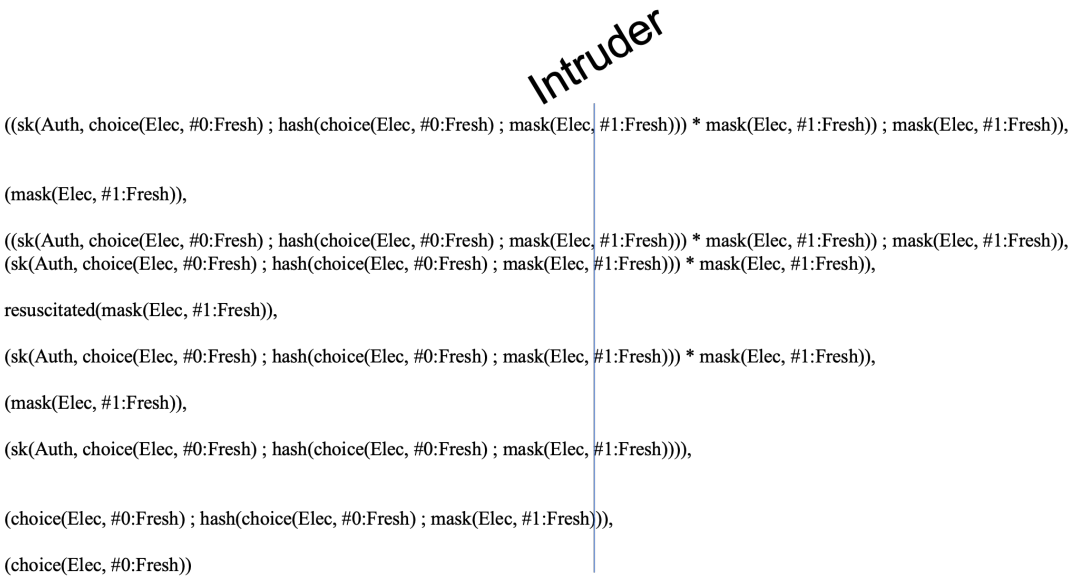

Ilustración 29. Pasos seguidos por el intruso en el ataque.

En este caso esa utilidad es inmediata; en el siguiente paso resucita la búsqueda utilizando el valor de la máscara para -implícitamente- hacer la inversa y con esta anular la operación multiplicación exponencial. Ha conseguido desenmascarar el grueso del mensaje.

A la información buscada solo le queda la protección que le proporciona la cifra/firma con la clave privada de la Autoridad de Identificación. Sin embargo, para el funcionamiento del protocolo se asume que la clave pública asociada a dicha clave privada es universalmente conocida.

El atacante puede ahora usar la clave pública relacionada, descifrar el bloque y obtener así el *Choice* y el *Hash*, que puede desagregar para despejar el *Choice* e incluso comprobar su validez.

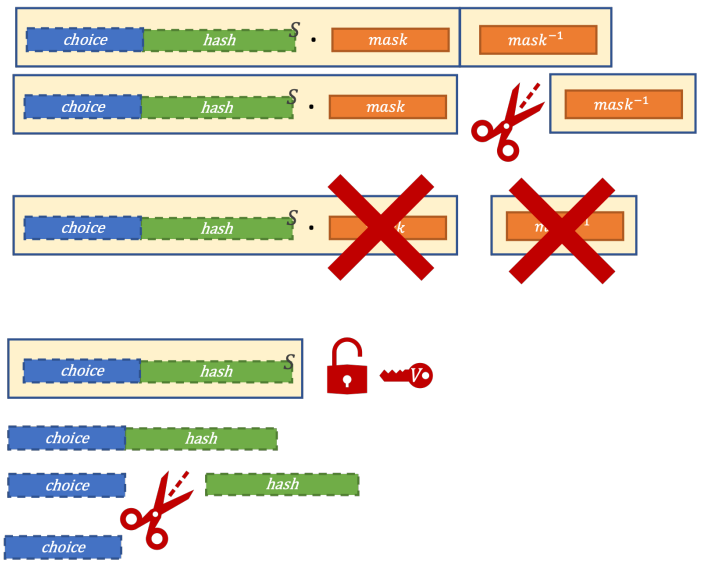

Ilustración 30. Interpretación gráfica de las operaciones del intruso.

En una interpretación rápida -gráfica- de estos resultados se aprecia que ante un medio de comunicación que no otorgue protección, el atacante solo ha tenido que imitar los pasos legítimos del protocolo para obtener el voto – e incluso comprobar su validez.

Es decir, si no se cumpliese el requisito de seguridad en el medio de transmisión, se confirmaría que el propio protocolo ya proporciona las herramientas para desvelar el sentido del voto. En concreto esto es necesario para el buen funcionamiento del rol del *Colegio Electoral Remoto*.

Sin embargo, gracias a la filosofía de segregación de los roles, de este ataque no se desprende la identidad del votante. El protocolo protege la identidad del elector fielmente.

6.3. Confidencialidad manteniendo la seguridad de los canales

En el punto anterior se ha comprobado qué si la privacidad del medio no está asegurada, la confidencialidad del sentido de voto no está asegurada. No así con la identidad del votante, que seguiría siendo desconocida para el atacante.

Este es el pilar fundamental vital sobre el que se ha creado el protocolo TAVS. Los valores principales por proteger son la identidad del elector y la relación de un elector específico con sus votos concretos. Desde un principio el objetivo ha sido conseguir mantener esa relación oculta «parcialmente» a cada una de las partes, y revelar solo la porción de información que necesitan para su rol. Se implementa el mecanismo de «Necesidad de conocer» de aplicación en los sistemas de información clasificada.

En este punto nos proponemos re-modelar el protocolo de manera que ahora la protección sea explícitamente incorporada al modelo. Para ello se utilizará el mecanismo de clave pública y privada.

```
eq STRANDS-PROTOCOL

  = :: Choice, Mask :: *** STRANDS DEL ELEC ***

  [ nil |
+(ELEC ; AUTH ; (pk(AUTH,((choice(ELEC,Choice) ; hash(choice(ELEC,Choice) ; mask(ELEC,Mask))) * pk(AUTH,mask(ELEC,Mask))))),
-(AUTH ; ELEC ; (pk(ELEC,(sk(AUTH,(choice(ELEC,Choice) ; hash(choice(ELEC,Choice) ; mask(ELEC,Mask)))) * mask(ELEC,Mask))))),
+(ELEC ; RPS ; (pk(RPS,(sk(AUTH,(choice(ELEC,Choice) ; hash(choice(ELEC,Choice) ; mask(ELEC,Mask)))) * mask(ELEC,Mask)) ; mask(ELEC,Mask)))),
   nil ] &

   :: nil :: *** STRANDS DEL IDENTIFICATION AUTHORITY ***

   [ nil |
-(ELEC ; AUTH ; pk(AUTH,((CH ; hash(CH ; Msk)) * pk(AUTH,Msk)))),
+(AUTH ; ELEC ; pk(ELEC,(sk(AUTH,(CH ; hash(CH ; Msk))) * Msk))),
nil ] &

   :: nil :: *** STRANDS DEL REMOTE POLLING STATION ***

   [ nil |
-(ELEC ; RPS ; pk(RPS,(sk(AUTH,(CH ; hash(CH ; Msk))) * Msk) ; Msk)),
   nil ]

 [nonexec] .
```

Ilustración 31. Strands de TAVS modificados para explicitar la protecciónde l medio de comunicación.

Se han modificado los *strands* -cambio marcado en negrita- para que cada transmisión, el paquete de datos útiles sean cifrados con la clave pública del receptor. Recordemos que, en los strands Dolev-Yao no se contemplaba la capacidad del atacante para descifrar mensajes cifrados con clave pública. Es decir, el intruso no conoce, ni puede usar las claves privadas de los participantes y por tanto los contenidos quedan fuera de su alcance.

```
eq ATTACK-STATE(3) *** MEDIO DE COMUNICACIÓN PROTEGIDO POR CLAVE ASIMÉTRICA ***

  = :: Choice, Mask :: *** EJECUCIÓN DEL ELEC ***

    [ nil,
  +(Elec ; Auth ; pk(Auth,((choice(Elec,Choice) ; hash(choice(Elec,Choice) ; mask(Elec,Mask))) * pk(Auth,mask(Elec,Mask))))),
  -(Auth ; Elec ; pk(Elec,(sk(Auth,(choice(Elec,Choice) ; hash(choice(Elec,Choice) ; mask(Elec,Mask)))) * mask(Elec,Mask)))),
  +(Elec ; Rps ; pk(Rps,(sk(Auth,(choice(Elec,Choice) ; hash(choice(Elec,Choice) ; mask(Elec,Mask)))) * mask(Elec,Mask)) ; mask(Elec,Mask)))
     | nil ]
     || choice(Elec,Choice) inI, empty
     || nil
     || nil
     || nil

  [nonexec] .
```

Ilustración 32. Attack-state modificado para explicitar la protección del medio de comunicación.

Similarmente, se reconstruye el *attack-state,* haciendo que los mensajes desde y hacia el *elector,* estén protegidos por la clave pública correspondiente. No se modifica la condición final que ha de satisfacer el atacante. Conocer el *choice*.

```
reduce in MAUDE-NPA : true .
rewrites: 0 in 0ms cpu (0ms real) (~ rewrites/second)
result Bool: true
Maude> red summary (3,0) .
reduce in MAUDE-NPA : summary(3, 0) .
rewrites: 7469763 in 13101ms cpu (13239ms real) (570158 rewrites/second)
result Summary: States>> 1 Solutions>> 0
Maude> red summary (3,1) .
reduce in MAUDE-NPA : summary(3, 1) .
rewrites: 79256 in 289ms cpu (294ms real) (273679 rewrites/second)
result Summary: States>> 0 Solutions>> 0
Maude> red summary (3,2) .
reduce in MAUDE-NPA : summary(3, 2) .
rewrites: 112 in 0ms cpu (0ms real) (235789 rewrites/second)
result Summary: States>> 0 Solutions>> 0
Maude>
```

Ilustración 33. Ejecución del attack-state modificado para explicitar la protección del medio de comunicación.

Se comprueba que Maude-NPA no es capaz de encontrar estados para la resolución del problema. Si le consultamos el detalle, indica que el conjunto está vacío.

```
reduce in MAUDE-NPA : initials(3,1) .
rewrites: 109 in 0ms cpu (0ms real) (~ rewrites/second)
result IdSystemSet: (empty).IdSystemSet
Maude>
```

Ilustración 34. Resultado de conjunto de estados vacío.

Este resultado confirma lo esperado. Bajo las condiciones esperadas de seguridad en el medio de comunicación, TAVS resulta seguro. El atacante no es capaz de generar operaciones sobre TAVS que conduzcan a revelación alguna. Para tener éxito, el atacante debería primero realizar un ataque al protocolo de cifra asimétrica utilizado.

Una de las lecciones que se pueden extraer de este resultado es la versatilidad y modularidad que proporciona TAVS al delegar aquellas partes que no son explícitamente del problema de la votación electrónica. Hemos visto como la propiedad de anonimato del elector depende directamente de TAVS. Por el contrario, la propiedad de confidencialidad del proceso -en realidad de la comunicación- se proporciona en una capa externa independiente del protocolo. Esta característica hace que TAVS pueda adaptarse fácilmente a la evolución las necesidades de cifrado, mientras se centra en su función primordial. Estamos ante un protocolo muy bien planteado.

6.4. Impostación del elector.

Se plantea en este apartado la posibilidad de que el intruso se interponga entre *Elector* e *Identity Authority en la petición del primero*. Este escenario es planteable si y solo si el medio de comunicación no es seguro.

```
eq ATTACK-STATE(3)

 = :: Choice, Mask :: *** EJECUCIÓN DEL AI ***
  [ nil,
                -(Elec ; Auth ; ((choice(Elec,Choice) ; hash(choice(Elec,Choice) ; mask(Elec,Mask))) * pk(Auth,mask(Elec,Mask)))),
                +(Auth ; Elec ; (sk(Auth,(choice(Elec,Choice) ; hash(choice(Elec,Choice) ; mask(Elec,Mask)))) * mask(Elec,Mask)))
  | nil ]
  || empty
  || nil
  || nil
  || never
  (:: R:FreshSet ::
  [nil |
                +(Elec ; Auth ; ((choice(Elec,Choice) ; hash(choice(Elec,Choice) ; mask(Elec,Mask))) * pk(Auth,mask(Elec,Mask)))),
                -(Auth ; Elec ; (sk(Auth,(choice(Elec,Choice) ; hash(choice(Elec,Choice) ; mask(Elec,Mask)))) * mask(Elec,Mask))),
  nil]
  & S:StrandSet || K:IntruderKnowledge)

[nonexec] .
```

Ilustración 35. Attack-pattern sobre la identificación del elector sin protección en el medio

Partiendo de tales condiciones: se permite al *Elector* enviar su petición de certificación a la Identity Authority; el *intruso* la interceptará y extraerá la solicitud del medio, modificándola con otra identidad y reenviándola a la *Identity Authority*. Este, validaría la identidad impostada y contestaría con la pre-papeleta certificada al remitente -intruso; e introduciría la pre-papeleta en la lista de revocación preventiva.

El *elector* legítimo se quedaría a la espera de una contestación que no le llegaría nunca, siéndole sustraído ese intento de voto. Técnicamente estaríamos ante un ataque a medio camino entre el secuestro de la sesión y la denegación de servicio. Esa pre-papeleta concreta estaría en la *Revoked Board* sin posibilidad de recuperarla si el atacante no lo desea, pues dicha pre-papeleta no incluye información de identidad alguna.

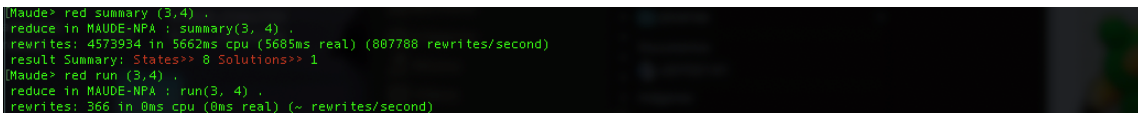

Ilustración 36. Solución encontrada para impostación del Elector.

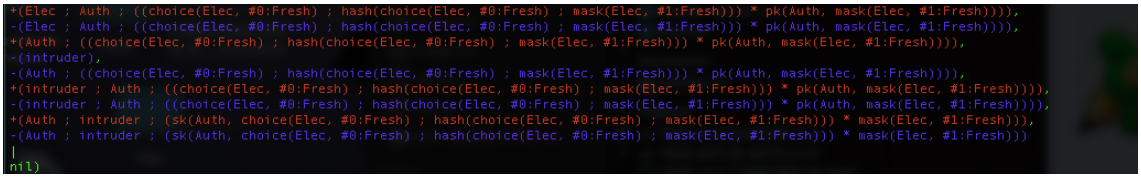

Ilustración 37. Resultado de la impostación del Elector.

Este ataque plantea más cuestiones que soluciones al atacante, por varias razones: Primero, el atacante habría de disponer de credenciales válidas para poder sustituir al *elector* real A. Segundo, el atacante no sabe el sentido del voto del *elector* A, y si lo reutiliza con otro *elector* B estará robando el derecho de voto este último, no al *elector* A. Tercero, el *elector* A podría regenerar otra *pre-papeleta* y reiniciar el proceso desde cero.

Pero, en fin, recordemos de nuevo que bajo las condiciones requeridas por TAVS, este ataque no sería viable. En cualquier caso, para evitar esta situación podría incluirse en el protocolo un valor temporal máximo, dentro del cual el elector debería mostrar interés en cerrar la validación, aceptando o rechazando la papeleta certificada por la IA.

6.5. Secuestro de sesión.

Por último, quedaría por comprobar la posibilidad de contestar al *elector* en nombre de la *Identity Authority*. Una vez más, es necesario asumir la relajación de la condición de seguridad del medio, que ya hemos utilizado.

El atacante dejaría llegar la petición inicial desde el *elector* hasta la *IA*, para a continuación capturar la respuesta de este.

El intruso podrá decidir entregar al *elector* el mensaje con la cabecera cambiada, afectando al proceso o bien negar la *pre-papeleta* certificada, secuestrando el intento de voto como veremos.

```
eq ATTACK-STATE(3) *** SANTI ***
  = :: Choice, Mask :: *** EJECUCIÓN DEL ELEC ***
    [ nil,
    +(Elec ; Auth ; ((choice(Elec,Choice) ; hash(choice(Elec,Choice) ; mask(Elec,Mask))) * pk(Auth,mask(Elec,Mask)))),
    -(Auth ; Elec ; (sk(Auth,(choice(Elec,Choice) ; hash(choice(Elec,Choice) ; mask(Elec,Mask)))) * mask(Elec,Mask)))
    | nil ]
    || (intruder ; Elec ; (sk(Auth,(choice(Elec,Choice) ; hash(choice(Elec,Choice) ; mask(Elec,Mask)))) * mask(Elec,Mask))) inI, empty
    || nil
    || nil
    || nil
  [nonexec] .

endfm
```

Ilustración 38. Attack-pattern de secuestro de sesión.

Un ataque del primer tipo no alterará el paquete interno que compone el voto. Pues este se encuentra protegido por la clave privada de la *Identification Authority*.

Solo modifica la cabecera, pero en un proceso electoral cualquier modificación sobre lo esperable sembraría dudas -razonables- y es intolerable. Obligará al *elector* a invalidar esa *pre-papeleta* y reiniciar el proceso.

Ilustración 39. Resultado del ataque para invalidar el proceso.

La otra opción que puede explotar el atacante consiste en que ha sustituido al *Elector* legítimo en el proceso de validación; y ha quedado en poder de una papeleta certificada anotada a nombre del *Elector* legítimo. El *Elector* no recibe la pre-papeleta certificada ni las transacciones de validación finales; por lo que solo puede anular su pre-papeleta de una forma: volviendo a generar desde cero el proceso. De esta manera la nueva papeleta invalidaría a la anterior en el *Revokation Board*.

Aunque el atacante desconozca el sentido del voto, ha sustraído temporalmente el derecho a voto del *Elector* afectado.

Hay que tener presente que TAVS resuelve los problemas inherentes al proceso electoral por sus características. No aborda - ni debe hacerlo - los problemas generales de la comunicación. Este ataque es lateral a TAVS y el daño provocado es colateral -como lo sería si se atacase desde otros protocolos.

Este resultado solo refuerza la necesidad de la condición de comunicación segura. Si esta se cumple, entonces el protocolo no se vería de manera alguna. Obviamente si implementamos la capa de cifrado del mensaje completo, el atacante queda incapaz.

7. Conclusiones

Al plantear el trabajo se buscaba confirmar o desmentir la validez de Maude-NPA en la validación formal de un protocolo de voto electrónico -en este caso TAVS.

Tras la experimentación realizada y los resultados obtenidos, ha quedado demostrado que es completamente viable utilizar Maude-NPA para la evaluación del protocolo TAVS.

Además, era también objetivo comprobar las propiedades del protocolo. Asegurar la integridad y la corrección del voto, la verificabilidad y auditoría, pero fundamentalmente certificar la resistencia a la coerción.

En este sentido, la ejecución del modelo estándar de TAVS corrobora el correcto funcionamiento del protocolo. La evaluación con el modelo relajado -asumiendo la inseguridad de las comunicaciones- ha expuesto los distintos caminos que el atacante podría llegar a tomar para afectar al proceso electoral.

Para contrarrestar esta debilidad forzada, se ejecutó de nuevo el ataque a la confidencialidad, pero introduciendo de manera representativa la capa adicional de seguridad antes ausente. El resultado fue plenamente satisfactorio, y se demostró la incapacidad del intruso para afectar al protocolo. Maude-NPA no encontró ninguna solución entre todas las posibles configuraciones posibles en su árbol de búsqueda inversa. Este resultado es obviamente extensible a todos los casos particulares posteriores y así debe entenderse

La simulación de los casos particulares -relajados- confirma que los requisitos planteados por los autores de TAVS son una base irrenunciable para el correcto funcionamiento del protocolo. Se ha verificado la integridad del modelo según lo planteado pues solo se han encontrado ataques cuando se han relajado los pre-requisitos.

8. Trabajos futuros

Este trabajo inicial con Maude-NPA sobre TAVS ha sido solo una pequeña aproximación, más prueba de validez del concepto de validación formal automática que auténtico estudio intensivo del protocolo. Se decidió simplificar el modelo considerablemente, eliminando partes que no aportaban valor añadido al núcleo del protocolo.

A partir de este momento, queda pendiente enriquecer el modelo del protocolo, introduciendo:

Representar de forma permanente la capa de confidencialidad en las comunicaciones y añadir la autenticación de los roles, añadiendo esta capa externa a la anterior.

Todo el mecanismo de aceptación de papeletas y gestión de errores entre el *Elector* y la *Identification Authority*, que no han sido consideradas en este trabajo. Es común que los fallos explotables de un protocolo se encuentren no en la parte fundamental, si no más bien en los rincones de la gestión. Con esto se disiparían posibles dudas sobre repetición de papeletas, denegación de servicio, secuestro del voto o no repudio por los participantes.

Además, una vez se ha demostrado la viabilidad de la evaluación automática, es el momento de desarrollar nuevos patrones de ataque, más elaborados, para demostrar sin lugar a duda de la fortaleza de TAVS.

9. Bibliografía

[1] Wikipedia, «Wikipedia,» 8 mayo 2022. [En línea]. Available: https://en.wikipedia.org/wiki/Electronic_voting_by_country#Internet_voting. [Último acceso: 14 junio 2022].

[2] A. M. Larriba, J. M. Sempere y D. Lopez, «A two authorities electronic vote scheme,» *Computers & Security,* vol. Computers & Security, nº 97, 2020.

[3] E. Santiago, M. Catherine y M. Jose, «Maude-NPA, Version 3.1,» Illinois, 2017.

[4] D. Basin, C. Cremers, J. Dreier y S. Mei, «Tamarin Prover,» [En línea]. Available: https://tamarin-prover.github.io/. [Último acceso: 7 junio 2022].

[5] federal Chancellery FCh, «E-Voting,» 2019. [En línea]. Available: https://www.bk.admin.ch/bk/en/home/politische-rechte/e-voting.html#:~:text=In%20Switzerland%2C%20e%2Dvoting%20means,electorate%20in%20over%20300%20trials.. [Último acceso: 14 junio 2022].

[6] Federal Chancellery FCh, «https://www.bk.admin.ch/bk/en/home/politische-rechte/e-voting.html#:~:text=In%20Switzerland%2C%20e%2Dvoting%20means,electorate%20in%20over%20300%20trials.,» 2020. [En línea]. Available: Redesign and relaunch of trials. [Último acceso: 14 junio 2022].

[7] J. Bannet, D. W. Price, A. Rudys, J. Singer y D. S. Wallach, "Hack-a-vote: Security issues with electronic voting systems," doi: 10.1109/MSECP.2004.1264851., vol. 2, New York: IEEE Security & Privacy, 2004, pp. 32-37.

[8] B. Schneier, «Schneier on Security,» 10 11 2004. [En línea]. Available: https://www.schneier.com/blog/archives/2004/11/the_problem_wit.html. [Último acceso: 7 junio 2022].

[9] A. Juels, D. Catalano y M. Jakobsson, Coercion-Resistant Electronic Elections, Berlin, Heidelberg : Springer, 2002.

[10] A. Acquisti, Receipt-free homomorphic elections and write-in ballots, Pittsburgh, Pensilvania: Carnegie Mellon University, 2004.

[11] P. Y. A. Ryan, D. Bismark, J. Heather, S. Schneider y Z. Xia, «The Pretˆ a Voter Verifiable Election System,» IEEE Transactions on Information Forensics and Security, New York, 2009.

[12] J.-M. Bohli, J. Mueller-Quade y S. Roehrich, Bingo Voting: Secure and coercion-free voting using a trusted random number generator, Berlin: Springer, 2007.

[13] N. S. Miramirkhani, R. Jalili y M. Yarmohamadi, «FOO e-voting protocol: Inductive analysis of the eligibility property,» de *9th International ISC Conference of Information Security & Cryptology*, Tabriz, Iran , 2012.

[14] V. Cortier, Formal verification of e-voting: solutions and challenges, vol. 2, Chausseestraße, Berlin: ACM SIGLOG News, 2015, p. 10.1145/2728816.2728823.

[15] B. Blanchet, «An efficient cryptographic protocol verifier based on prolog rules,» de *Proceedings. 14th IEEE Computer Security Foundations Workshop*, Cape Breton, NS, Canada, 2001.

[16] B. Chiara, B. Mikael, D. Pierpaolo, N. Flemming y H. Riis, Static validation of security protocols, vol. 13, Amsterdam: Journal of Computer Security, 2005, p. 347–390.

[17] V. Cortier, F. Eigner, . S. Kremer, M. Maffei y C. Wiedling, «Type-Based Verification of Electronic Voting Protocols,» de *Principles of Security and Trust*, Berlin, Heidelberg, Springer, 2015.

[18] S. Delaune, S. Kremer y M. Ryan, Verifying privacy-type properties of electronic voting protocols, vol. 17, Berlin: Journal of Computer Security, 2009, p. 435–487.

[19] S. Kremer y M. Ryan, «Analysis of an Electronic Voting Protocol in the Applied Pi Calculus,» de *Programming Languages and Systems. ESOP 2005. Lecture Notes in Computer Science*, Berlin, Heidelberg, Springer, 2005.

[20] J. Lluch Palop, «Verificación automática del protocolo TLS 1.3 usando Maude-Npa,» TFM, Máster Universitario en Ingenieria y Tecnología de Sistemas Software DSIC, Valencia, 2019.

[21] B. Blanchet y V. Cheval, «ProVerif: Cryptographic protocol verifier in the formal model,» [En línea]. Available: https://bblanche.gitlabpages.inria.fr/proverif/.

[22] R. Rivest, A. Shamir y L. Adleman., A Method for Obtaining Digital Signatures and Public-Key Cryptosystems, vol. 21, Massachusetts Institute of Technology: Massachusetts, 1978, pp. 120-126.

[23] D. Dolev y A. C. Yao, *On the security of public key protocols,* Vols. %1 de %2 IT-29 (2), Stanford University Stanford, CA 94305: IEEE Transactions on Information Theory, 1983.

10. Glosario

Attack-pattern: Descripción del comportamiento del protocolo y del ataque del intruso en el que se define el objetivo final de este.

Ballot: Papeleta certificada y válida para ser utilizada en el proceso electoral.

Choice: Abstracción para representar el sentido del voto codificado. Es independiente de la tecnología y del tipo de elección.

DRE: Direct Recording Electronic

Duoble vote: Situación en la que se produce una utilización repetida de un voto.

IA: Autoridad de Identificación. Valida ciegamente las pre-papeletas contrastando la validez de la identidad solicitante. Genera papeletas certificadas válidas..

Mask: Campo de la papeleta (y pre-papeleta) que enmascara matemáticamente el sentido del voto antes hasta ser entregado al Colegio Electoral Remoto.

Maude-NPA: Lenguaje de programación Maude - Naval Research Laboratory Protocol Analyzer.

Pre-ballot: Pre-paleta con el sentido del voto enmascarado matemáticamente, emitida por el elector y enviada a la IA para su certificación.

RB:Revoked Board. Panel público en el que se anotan las papeletas certificadas hasta ser aceptadas por su elector correspondiente. Evita que se haga mal uso de ellas hasta estar confirmadas.

RBB: Remote Bulletin Board. Pane público en el que se publican los hash de cada papeleta de manera que el elector pueda comprobar que se ha contado su voto correctamente.

RPS: Remote Polling Station. Colegio Electoral Remoto.

TAVS: Acrónimo del protocolo «Two authorities vote scheme» de voto propuesto por Larriba, Sempere y López en [2].

11. Anexo I. Objetivos de Desarrollo Sostenible.

11. ANEXO I

OBJETIVOS DE DESARROLLO SOSTENIBLE

Grado de relación del trabajo con los Objetivos de Desarrollo Sostenible (ODS).

Objetivos de Desarrollo Sostenibles	Alto	Medio	Bajo	No Procede
ODS 1. **Fin de la pobreza.**				X
ODS 2. **Hambre cero.**				X
ODS 3. **Salud y bienestar.**			X	
ODS 4. **Educación de calidad.**				X
ODS 5. **Igualdad de género.**				X
ODS 6. **Agua limpia y saneamiento.**				X
ODS 7. **Energía asequible y no contaminante.**				X
ODS 8. **Trabajo decente y crecimiento económico.**				X
ODS 9. **Industria, innovación e infraestructuras.**		X		
ODS 10. **Reducción de las desigualdades**.		X		
ODS 11. **Ciudades y comunidades sostenibles**.		X		
ODS 12. **Producción y consumo responsables.**				X
ODS 13. **Acción por el clima.**		X		
ODS 14. **Vida submarina.**				X
ODS 15. **Vida de ecosistemas terrestres.**				X
ODS 16. **Paz, justicia e instituciones sólidas.**	X			
ODS 17. **Alianzas para lograr objetivos.**				X

Reflexión sobre la relación del TFG/TFM con los ODS y con el/los ODS más relacionados.

La relación directa entre este trabajo de Fin de Máster y los ODS se plantea a través del beneficio inmediato que se deriva del fortalecimiento de los procesos electorales.

Es un requisito fundamental de toda sociedad libre y avanzada el dotarse de un gobierno legítimo y transparente. La primera piedra de ese edificio son unos procedimientos electorales seguros, libres y limpios; que aseguren la participación en igualdad y sin riesgos por parte del elector.

Un mecanismo de voto electrónico que cumpla con los requisitos de seguridad, usabilidad y limpieza proporciona la posibilidad de aumentar el número y calidad de consultas realizables al:

- Abaratar el coste logístico y económico de la consulta.
- Permitir la participación deslocalizada de un espacio físico restrictivo.
- Permitir la participación desligada en un periodo temporal más amplio.
- Eliminar las aglomeraciones y colas de espera físicas.
- Plantear consultas más detalladas.
- Obtener resultados de cuasi manera inmediata.

Todos estos puntos se traducirían en una ampliación de la participación ciudadana, tanto horizontalmente (más consultas) como verticalmente (consultas sobre temas más específicos y detallados), que se traducen en una consolidación de la democracia participativa.

En sistemas estables (y con un buen funcionamiento) como el español es difícil argumentar que un sistema de voto electrónico vaya a mejorar la igualdad entre los españoles a la hora de votar. Al fin y al cabo, nuestro modelo funciona de manera sólida y eficiente. Sin embargo, se presentan casos en los que el acceso a las elecciones aun constituye problemas, generalmente logísticos. Pensemos en dotaciones de buques en alta mar, personal desplazado por motivos laborales fuera del territorio nacional o simplemente trabajadores esenciales que deben abandonar sus puestos para ejercer el derecho a voto. En todos estos casos el voto electrónico mejoraría su proceso.

Además, al permitir la ampliación y profundización de los temas sometidos a consulta pública, los ciudadanos ocuparían una posición central en el gobierno real. La democracia podría permitirse reducir el porcentaje de «representatividad» para aumentar el porcentaje de «directa» que tiene. El modelo sería implementable con carácter general en todos los niveles de la administración; de tal manera que podamos, no solo elegir a los congresistas y senadores nacionales y autonómicos o al alcalde y

concejales, si no también decidir asuntos mucho más pragmáticos: sí o no a peatonalizar cierta calle de nuestra ciudad o disponer de una parte del presupuesto entre una u otras opciones.

Obviamente, un buen sistema de elecciones no garantiza que un gobierno legítimamente elegido puede ser un mal gobierno en sus actos. Pero una alta participación por parte del cuerpo electoral fuerza probabilísticamente que las decisiones (los resultados electorales) se alejen de opciones «a priori» catastróficas.

ETS Enginyeria Informàtica
Camí de Vera, s/n. 46022. València
T +34 963 877 210
F +34 963 877 219
etsinf@upvnet.upv.es - www.inf.upv.es