Firma electrónica de contratos

Josep Lluís Ferrer Gomila
Llorenç Huguet Rotger
M. Magdalena Payeras Capellà

PID_00199789

UOC
Universitat Oberta de Catalunya

Índice

Introducción

El ciclo de vida de un proceso de contratación comprende distintas fases, entre las que podemos encontrar: negociación, firma del contrato (perfeccionamiento) y ejecución del contrato. La primera y última no son objeto de esta sección, y en general no requieren elevadas medidas de seguridad para ser llevadas a cabo de manera electrónica. En este módulo nos centramos en la fase de firma del contrato. Se entiende que un contrato se ha perfeccionado (término jurídico que significa que se da por firmado el contrato) cuando las partes implicadas se han comprometido a la ejecución del mismo.

Desde la perspectiva técnica, la firma electrónica de contratos forma parte de un conjunto de problemas que de manera genérica recibe el nombre de intercambio equitativo de valores. La idea sustancial es que dos o más partes tienen un objeto (uno cada una de ellas) que quieren intercambiar por el de las otras partes intervinientes en la transacción, pero quieren tener la garantía de que si ellos proporcionan su objeto, recibirán el objeto o objetos que esperan a cambio. Llegamos así, de forma rápida, a la propiedad fundamental que pediremos a cualquier protocolo para la firma electrónica de contratos: la equidad. Es decir, que o todas las partes implicadas reciben lo que esperan al final del intercambio, o ninguna de ellas estará en una posición ventajosa al final de ese intercambio.

> La propiedad fundamental de los protocolos de firma electrónica de contratos es la equidad: que ninguna de las partes se encuentre en una situación de desventaja tras finalizar la ejecución del protocolo de firma de contratos.

La solución más habitual en el mundo en papel es casi trivial, pues muchas firmas de contratos se realizan de manera presencial, y eso garantiza que las partes obtengan lo que están esperando, sin asumir el riesgo de proporcionar la suya sin recibir la de la contraparte. Obviamente, en el mundo electrónico el intercambio cara a cara no es posible, y por tanto deberemos proporcionar soluciones alternativas pero que permitan mantener (o mejorar) los niveles de seguridad del mundo en papel.

Las soluciones para la firma electrónica de contratos suelen clasificarse en función de si interviene o no una tercera parte de confianza (TTP, del inglés por *Trusted Third Party*), y si interviene, cómo lo hace (en todos los contratos, si surgen problemas, etc.). Por ello, expondremos soluciones con TTP y soluciones sin TTP.

1. Soluciones para la firma electrónica de contratos con TTP

La clasificación realizada en la introducción respecto de las TTP puede refinarse más, según el grado de intervención de la TTP en la ejecución de los protocolos para la firma electrónica de contratos. Así hablamos de protocolos para la firma electrónica de contratos con:

- TTP *in-line*, cuando la TTP interviene en todos los pasos de la ejecución del protocolo
- TTP *on-line*, cuando la TTP interviene en todas las ejecuciones del protocolo de firma electrónica de contratos, pero no en todos los pasos del mismo
- TTP *off-line*, cuando la TTP solo interviene en caso de que surjan conflictos o problemas entre las partes, es decir, bajo demanda de una o más de ellas; por tanto, en este tipo de protocolos la TTP no interviene de manera genérica, y también reciben el nombre de protocolos optimistas

Ahora nos centraremos en la firma electrónica de contratos entre dos partes, es decir, dos partes contratantes quieren intercambiar sus firmas sobre el texto de un contrato.

Antes de empezar con la descripción de los protocolos, procederemos a describir la notación para designar a los actores que usaremos. Los actores que intervendrán son, por una parte, los firmantes del contrato que denominaremos A(lice) y B(ob), y por otra parte tenemos la tercera parte de confianza que denominaremos T.

Solución con TTP *in-line*

Empezaremos con la que se puede considerar como la solución más sencilla posible, utilizando una TTP *in-line*. A y B deben enviar el contrato y su firma sobre el mismo a la TTP. Una vez la TTP dispone de las dos copias firmadas, procederá a verificar que las firmas son correctas y, si es el caso, retransmitirá la copia firmada por a A a B, y la de B a A:

A $\rightarrow$ T: M,$Sign_A(M)$
B $\rightarrow$ T: M,$Sign_B(M)$
T : verifica las firmas de A y B

T $\rightarrow$ A: M,$Sign_B(M)$
T $\rightarrow$ B: M,$Sign_A(M)$

Obsérvese que en este sencillo protocolo la TTP es la que proporciona las garantías de que el intercambio sea equitativo. Hasta que no dispone de las copias de las dos partes, no las retransmite. Por tanto, se cumple la propiedad de equidad. No obstante, con este sencillo ejemplo detectamos algunos inconvenientes de este tipo de soluciones.

Por una parte, el hecho de que la TTP deba intervenir en cada intercambio puede suponer un problema de cuello de botella por lo que se refiere a las comunicaciones con ella, o en cualquier caso, su intervención permanente puede encarecer el proceso de contratación. Por ello suelen preferirse aquellas soluciones que no requieren de una participación en todas las ejecuciones de la TTP.

Otro aspecto que debe considerarse es la confianza que debe depositarse en la TTP, como mínimo desde dos puntos de vista. Por una parte es fácil observar que la TTP puede favorecer a una de las partes, proporcionando la copia, por ejemplo, de B a A y no la de A a B. Esto conduciría a una situación no equitativa por culpa de la TTP.

Por otra parte, tal como se ha diseñado el protocolo, la TTP tiene acceso a toda la información relacionada con el contenido del contrato, por tanto, la privacidad y el secreto de la transacción queda también en manos de la TTP, que podrá difundirla a terceros sin el conocimiento de A y B.

Solución con TTP *off-line* asíncrona

Para resolver algunos de los problemas indicados anteriormente, explicaremos a continuación una solución optimista o con TTP *off-line*. En esta solución aparecerán tres subprotocolos: un subprotocolo de intercambio, uno de cancelación y uno de finalización. Empezaremos explicando el de intercambio. Los pasos que deben seguirse son los siguientes:

A → B: M,$Sign_A(M)$
B → A: $Sign_B(M)$
A → B: $Sign_A[Sign_B(M)]$

Una vez intercambiada la anterior información, el contrato está firmado. Para A la prueba del contrato es sencillamente la firma de B sobre el mismo, y para B la firma del contrato consiste en la firma de A y el acuse de recibo (el tercer paso) que le ha enviado A. Obsérvese que si las dos partes actúan correctamente (y no hay problemas de comunicaciones) la TTP no tiene que intervenir.

En una solución optimista o con TTP *off-line*, si las partes actúan correctamente y no se producen problemas de comunicaciones, se consigue la propiedad de equidad sin que deba intervenir la TTP.

Pero, ¿qué sucede si alguna de las dos partes intenta hacer trampas? Más concretamente, ¿qué sucede si A, une vez que dispone de la firma de B, no envía el acuse de recibo del tercer paso? Si A actuara de esta manera, ella tendría copia firmada del contrato de B, y B solo dispondría de la evidencia de que A quería firmar el contrato, pero no de la evidencia plena de que el contrato está firmado. Por tanto, la situación no sería equitativa. Por ello es necesario un subprotocolo para que, si se da esta situación, le permita a B restablecer la equidad del intercambio. Se trata del subprotocolo de finalización:

B → T: M,$Sign_A(M)$,$Sign_B(M)$
T → B: $Sign_T[Sign_B(M)]$

En primer lugar B debe proceder a enviar a la TTP la información que le proporcionó A en el primer paso. Con ello (comprobando la firma realizada por A) la TTP puede verificar que A le propuso a B firmar el contrato M. Siendo así, y si la TTP no dispone de más información, la TTP le envía a B un acuse de recibo equivalente al que debería haber enviado A (pero ahora firmado por la TTP). De esta manera hemos restablecido la equidad para B.

Pero a continuación podemos observar que puede darse una situación no deseable para A. Con el subprotocolo anterior hemos conseguido que B, una vez ha recibido el primer paso de A, pueda finalizar la ejecución del protocolo de firma de contratos en cualquier momento, incluso si no envía (sin ni siquiera hacer el intento) la información del segundo paso a A.

A priori esto no supone un problema grave de equidad para A, porque si B no contacta con la TTP, ninguno de los dos tiene nada que comprometa a la otra parte, y si B contacta con la TTP, lo único que debe hacerse es permitir que A también pueda contactar con la TTP, la cual debería proporcionarle la firma de B (que este le había remitido a la TTP cuando contactó con ella):

A → T: M,$Sign_A(M)$
T → A: $Sign_B(M)$

Pero en el caso de que B no haya contactado con la TTP en el momento en que lo haga A, ¿qué debe hacer la TTP? Una primera opción sería que no hiciese nada, pero de esta manera A quedaría en una situación no deseable: A sabe que B puede contactar con la TTP en cualquier momento y obtener el acuse de recibo de la TTP. Para solucionar esta situación conflictiva caben dos soluciones.

En primer lugar podemos establecer una fecha de expiración tras la cual la TTP no resolverá nuevas peticiones, o bien puede añadirse una respuesta nueva de la TTP para cancelar el intercambio:

A → T: M,$Sign_A(M)$
T → A: $Sign_T(cancel - firma - M)$

Es decir, que si la TTP no ha recibido ninguna petición de B (por tanto, la TTP piensa que de momento para B todo es correcto), lo que debe hacer es enviar un mensaje de cancelación vinculado al contrato con A. Ahora deberíamos combinar los dos casos anteriores en un único subprotocolo para A:

A → T: M,$Sign_A(M)$
Si B_ha_contactado_con_T then T → A: $Sign_B(M)$
Else T → A: $Sign_T(cancel - firma - M)$

De esta manera contemplamos las dos posibles situaciones que pueden darse cuando A contacta con la TTP. Quedaría ahora por completar el subprotocolo de finalización que habíamos explicado para B. Ahora debe contemplarse el caso de que A haya contactado con la TTP antes que B:

B → T: M,$Sign_A(M)$,$Sign_B(M)$
Si A_ha_contactado_con_T then T → B: $Sign_T(cancel - firma - M)$
Else T → B: $Sign_T[Sign_B(M)]$

Ahora el protocolo de manera global cumple la propiedad fundamental de equidad. Para comprobarlo, analicemos las posibles situaciones que pueden darse:

- A y B finalizan la ejecución del subprotocolo de intercambio. En este caso queda claro que tras ejecutar los tres pasos, ambas partes acaban obteniendo prueba de que el contrato está firmado.
- A envía el primer mensaje y no recibe nada de B (porque B no ha enviado la respuesta, o porque se ha perdido mientras estaba en tránsito). A debe contactar con la TTP y si B no había contactado con T (eso significa que B no puede tener el acuse de recibo), recibirá un mensaje de cancelación; por tanto, ninguna de las dos partes tendrá prueba de que el contrato está firmado (porque de hecho no está firmado). En el caso de que B ya hubiese contactado con la TTP, esta le había proporcionado un acuse de recibo a B, y ahora debe proporcionar la firma de B a A: por tanto, ambas partes tendrán prueba de que el contrato está firmado.
- B envía su mensaje y no recibe el acuse de recibo. De manera análoga a la anterior, puede demostrarse que o ambos reciben una cancelación, o ambos dispondrán de una prueba de que el contrato está firmado.

Ya hemos demostrado que el protocolo anterior es equitativo, pero ahora explicaremos una situación aparentemente contradictoria que puede darse. Supongamos que se ejecuta el protocolo de intercambio sin ningún problema, pero además *a posteriori* A contacta con la TTP para cancelar el intercambio. Como la TTP no dispone de más información (B no ha contactado con ella), debe proporcionar un mensaje de cancelación a A. De esta manera A dispone de una doble prueba: que el contrato está firmado (con la prueba aportada por B) y que el contrato está cancelado (con la prueba aportada por T).

Pero en el caso de que surjan litigios entre A y B, para un árbitro externo quedará claro que A es tramposo si intenta alegar que el contrato está cancelado. Si A intenta hacer prevalecer la prueba de cancelación delante de un árbitro, B aportará los dos mensajes enviados por A (la firma de A y el acuse de recibo de A). En concreto, el acuse de recibo de A demuestra que este es tramposo, pues en ningún caso A habría enviado ese acuse de recibo sin haber dispuesto con anterioridad de la firma de B. En conclusión, A contactó con la TTP después de haber recibido la copia firmada de B, intentando hacer trampas.

Uno de los problemas que planteaba la solución con TTP *in-line* era que debía depositarse mucha confianza en la TTP. Deberíamos preguntarnos ahora cómo es el nuevo protocolo desde el punto de vista de posibles alianzas de la TTP con las distintas partes. Técnicamente denominamos verificabilidad de la TTP a esta propiedad.

Diremos que un protocolo de firma electrónica de contratos con TTP cumple la propiedad de que la TTP es verificable, si las partes intervinientes pueden demostrar un mal comportamiento de la TTP si esta no sigue los pasos previstos en el protocolo.

Para realizar el análisis del protocolo anteriormente propuesto, observaremos los dos posibles ataques que puede llevar a cabo la TTP. Una asunción que realizaremos es que la TTP siempre responde a las peticiones de las partes (pues de lo contrario sería posible demostrar ese mal comportamiento de la TTP con un árbitro externo).

El primer ataque que se podría realizar es una alianza de la TTP con A. Así, el objetivo sería proporcionar copia del contrato a A, y dejar sin prueba a B. Pero esta situación no es posible porque si B contacta con la TTP, esta o bien deberá proporcionarle el acuse de recibo de A, o el acuse de recibo de T, o el mensaje de cancelación. Si se dan las dos primeras situaciones, el ataque ha fracasado. Y si se da la tercera situación, B dispondrá de una evidencia que le permitirá demostrar *a posteriori*(si A intenta hacer valer la copia firmada de B) que o A o T hicieron trampas.

El segundo ataque que podría producirse es la alianza de T con B. En este caso, el objetivo es proporcionar copia del contrato a B, e intentar dejar sin pruebas a A. Para ello, la TTP proporciona el acuse de recibo de A y el de cancelación a B, y un mensaje de cancelación a A. Esta situación no puede ser detectada *a priori* por A. En este momento, B se encuentra en una situación de privilegio porque si *a posteriori* le interesa afirmar que el contrato está firmado, aportará el acuse de recibo de A, y en caso contrario sólo aportará la cancelación de T. Por el contrario, desde el punto de vista de A, el contrato está cancelado. ¿Cómo puede darse esta situación? Porque recordemos que existía la posibilidad de que tras finalizar el intercambio A contactara con la TTP para obtener un mensaje de cancelación e intentar hacer trampas. Esta posible situación ahora se ha vuelto en su contra.

También decíamos en el protocolo con TTP *in-line*, que las partes debían depositar mucha confianza en la TTP porque tenía acceso al contenido del mensaje. En la nueva propuesta hemos mejorado el problema un poco, pero no totalmente. Si la TTP no debe intervenir, no tendría acceso al mensaje, pero si la TTP debe intervenir, observamos que las partes envían el texto del mensaje, M, a la TTP.

Un protocolo para la firma electrónica de contratos cumple con la propiedad de confidencialidad si solo los firmantes del contrato tienen acceso al contenido del mismo, y ni tan siquiera la TTP tiene acceso a ese contenido.

Por tanto, el protocolo planteado es mejorable desde el punto de vista de la confidencialidad de la información intercambiada, porque no solo la TTP puede llegar a tener acceso a la información del contrato, sino también cualquier espía que tenga acceso al canal de comunicaciones (el contrato M se transmite en claro entre A y B). Vayamos a introducir unas pequeñas mejoras para conseguir la propiedad de confidencialidad. Lo primero que haremos es cifrar la comunicación entre A y B (esto se podría conseguir de forma sencilla en la práctica utilizando protocolos ya estandarizados como SSL/TLS):

A $\rightarrow$ B: $E_k(M)$,$PU_B(k)$,$Sign_A(M)$
B $\rightarrow$ A: $Sign_B(M)$
A $\rightarrow$ B: $Sign_A[Sign_B(M)]$

El mecanismo utilizado es sencillo y consiste en cifrar el contrato M con un criptosistema simétrico (por ejemplo, AES) con una clave secreta generada por A. Pero B necesitará conocer esa clave secreta, y por ello ciframos la clave secreta, k, con la clave pública de B de un criptosistema de clave pública (por ejemplo, RSA). Cuando B recibe la información del primer paso, la primera operación que debe realizar es un descifrado con su clave privada para recu-

perar la clave k. Con esta clave k puede realizar el descifrado del criptograma que contiene el texto del contrato. Finalmente, con el texto del contrato y la firma de A sobre este, B puede verificar si dicha firma es correcta.

Una vez resuelta la confidencialidad ante terceros, quedaría resolver la confidencialidad ante la TTP. En realidad, la TTP no necesita estrictamente tener acceso al contenido del contrato, pues la TTP solo necesita verificar que las firmas de las partes son correctas. Por ello es suficiente que le proporcionemos a la TTP el resumen del contrato que se ha utilizado para realizar la firma electrónica:

A → T: $H(M)$,$Sign_A(M)$
Si B_ha_contactado_con_T then T → A: $Sign_B(M)$
Else T → A: $Sign_T(cancel - firma - M)$

B → T: $H(M)$,$Sign_A(M)$,$Sign_B(M)$
Si A_ha_contactado_con_T then T → B: $Sign_T(cancel - firma - M)$
Else T → B: $Sign_T[Sign_B(M)]$

Para realizar la verificación de si las firmas son correctas, lo único que debe hacer la TTP es aplicar las claves públicas de A y B (según proceda) y a continuación comparar el valor obtenido con el resumen que se le ha transmitido (si coinciden, la firma es correcta, y en caso contrario, la firma no se corresponde con el resumen recibido). Ahora sí podemos afirmar que el protocolo cumple la propiedad de confidencialidad.

Una propiedad interesante del protocolo que acabamos de presentar es que no se han introducido restricciones temporales. Las partes pueden contactar con la TTP cuando lo deseen para finalizar la ejecución del protocolo. Por ello decimos que el protocolo es asíncrono (no requiere de la sincronización de los relojes de ninguna de las partes que intervienen en el mismo).

Un protocolo de firma electrónica de contratos cumple la propiedad de temporalidad (*timeliness* en inglés), si las partes pueden tener la garantía de que la ejecución del protocolo puede finalizarse cuando ellas deseen, sin asumir el riesgo de perder la equidad del intercambio.

Solución con TTP *off-line* síncrona

Hemos visto que el hecho de que el protocolo fuera asíncrono (las partes podían contactar con la TTP en cualquier momento), obligaba a introducir un subprotocolo de cancelación para A. Vayamos a ver ahora cómo podríamos

modificar la propuesta si establecemos un tiempo límite, es decir, que transcurrido un cierto tiempo ya sólo puedan hacerse consultas a la TTP de cuál es el estado de la transacción, pero no pedirle que realice ninguna acción de modificar el estado del intercambio. Supondremos que el subprotocolo de intercambio coincide con el anterior:

A → B: M,$Sign_A(M)$
B → A: $Sign_B(M)$
A → B: $Sign_A[Sign_B(M)]$

Tal como sucedía antes, puede darse la situación de que B envíe su compromiso, pero no reciba el acuse de recibo de A. Por ello debemos establecer un subprotocolo para que B pueda restablecer la equidad del intercambio. Pero ahora le imponemos una condición adicional a B, y es que debe contactar la TTP antes de un determinado valor temporal t. Si se encuentra en una situación no equitativa y no contacta la TTP antes de t, su problema ya no tendrá solución: quedará en desventaja ante A (pero debido a su inacción). Por tanto, el subprotocolo queda como sigue:

B → T: M,$Sign_A(M)$,$Sign_B(M)$
Si t-actual<t then T → B: $Sign_T[Sign_B(M)]$
Else T → B: $Sign_T[fuera-de-plazo]$

En este caso no tiene sentido que A contacte con la TTP para cancelar el intercambio, pues A sabe que si no recibe la firma de B y tras transcurrir el tiempo t (que ella ha elegido), podrá consultar la TTP para saber el estado del intercambio. El subprotocolo para A sería:

A → T: M,$Sign_A(M)$
Si B_ha_contactado_con_T then T → A: $Sign_B(M)$
Else T → A: $Sign_T(cancel-firma-M)$

De hecho, en el caso de que B no hubiese contactado con la TTP, no haría falta que T le enviase un mensaje de cancelación a A, porque tras el tiempo t, B ya no podrá conseguir el acuse de recibo de A ni de T, por tanto, ninguna de las partes tendrá nada que comprometa a la otra.

La introducción de una restricción temporal simplifica el protocolo planteado (y las acciones que debe realizar la TTP), pero puede suponer un inconveniente para las partes contratantes (en este caso para B). Por una parte, B debe tener su reloj sincronizado con la TTP, pues de lo contrario asume el riesgo de ver rechazada su petición por haber llegado fuera de plazo. Por otra parte, aunque los relojes estuvieran sincronizados, en el caso de que surjan problemas con el canal de comunicaciones, la equidad del intercambio puede verse comprometida para B, por la imposibilidad de contactar con la TTP dentro del plazo establecido.

Otra solución con TTP *off-line*

Llegado a este punto, en el que ya se han presentado dos soluciones (una síncrona y la otra asíncrona) para la firma electrónica de contratos, podría plantearse la cuestión de por qué un protocolo con tres pasos: ¿sería posible un protocolo en dos pasos? La respuesta es que no, porque la TTP no tendría manera de tomar una decisión que garantizara la equidad a la parte que esté actuando de manera honesta.

Por otra parte, podría pensarse en una solución en 4 pasos. Obviamente, la eficiencia disminuiría, pero tal vez se conseguiría alguna mejora respecto al protocolo de tres pasos. Un protocolo de 4 pasos sería de la siguiente manera:

A → B:M,$Sign_A(M)$
B → A: $Sign_B(M)$
A → B: $Sign_A[Sign_B(M)]$
B → A: $Sign_B(Sign_A[Sign_B(M)])$

Puede observarse que se ha conseguido cierta simetría entre las dos partes que deben firmar el contrato. Ahora la prueba para A es la firma de B y el acuse de recibo de B, tal como para B la prueba es la firma de A y el acuse de recibo de A. Quedaría ahora por ver cómo quedan alterados los subprotocolos para los casos de conflicto.

Obsérvese que ahora el que rápidamente puede quedar en desventaja es A (a diferencia de antes), porque una vez que A ha enviado su acuse de recibo queda en manos de B. Por ello debemos establecer un protocolo de finalización para A con la TTP:

A → T: M,$Sign_A(M)$,$Sign_B(M)$,$Sign_A[Sign_B(M)]$
T → A: $Sign_T(Sign_A[Sign_B(M)])$

De manera análoga a lo que hemos explicado en el primer protocolo optimista, ahora sería B el que podría quedar en una situación no equitativa, si no diseñamos un subprotocolo de finalización para él. Podría darse el caso de que A hiciera trampas, y sin haber enviado el tercer mensaje contactase con la TTP. De esta manera A consigue las pruebas que necesita (la firma de B y el acuse de recibo de T) y B solo tiene la firma de A que no es suficiente. Por ello necesitamos el siguiente subprotocolo:

B → T: M,$Sign_A(M)$,$Sign_B(M)$
T → B: $Sign_T(Sign_A[Sign_B(M)])$

Ahora deberíamos decidir si queremos que la solución sea síncrona (estableciendo un plazo para poder contactar con la TTP), o si deseamos una solución asíncrona (para lo que deberíamos diseñar un subprotocolo de cancelación

para B, con el objeto de que no deba esperar indefinidamente a ver qué hace A).

Este protocolo en cuatro pasos (tanto la versión síncrona como la asíncrona) introduce una mejora respecto de las versiones en tres pasos, y es que el comportamiento de la TTP es verificable, es decir, que si la TTP intenta hacer trampas, esta situación podrá ser detectada y demostrada.

2. Soluciones para la firma electrónica de contratos sin TTP

Si hemos dicho que las soluciones con TTP *on-line* o *in-line*, no son convenientes porque la TTP puede convertirse en un cuello de botella por lo que respecta a las comunicaciones, o no queremos asumir el coste que puedan comportar, la misma crítica, aunque en menor grado, puede dirigirse a las soluciones con TTP *off-line*. Por ello cabe preguntarse si existen propuestas de soluciones sin TTP. La respuesta es que sí, y que uno de los tipos de soluciones sin TTP se basa en lo que se conoce como el intercambio gradual de secretos. De forma sencilla podemos decir que la idea consiste en intercambiar la firma sobre el contrato a trozos (a continuación veremos que esto es una simplificación un poco burda).

Protocolo de cara o cruz

Para explicar un ejemplo concreto de protocolo basado en el intercambio gradual de secretos, primero debemos explicar un protocolo para jugar a cara o cruz a traves de las redes de comunicaciones. Para ello suponemos que dos usuarios quieren jugar al juego de cara o cruz en versión electrónica y uno de ellos, el iniciador (Alice), dispone de un par de claves de criptografía asimétrica, PR_1-PU_1 y PR_2-PU_2. En primer lugar remite las componentes públicas al otro jugador (Bob):

$$\text{A} \rightarrow \text{B: } PU_1, PU_2$$

A continuación B genera una clave de criptografía simétrica (k), y escoge de manera aleatoria una de las dos claves públicas que recibió procedentes de A, que la etiquetaremos como PU_j. Con la clave pública escogida cifrará la clave k, y transmitirá el resultado a A:

$$\text{B} \rightarrow \text{A: } PU_j(k)$$

Ahora A debe escoger de manera también aleatoria (no sabe cuál escogió B) una de las dos claves privadas. Supongamos que escoge PR_1 y realiza el descifrado con esta clave de lo que ha recibido de B:

$$X = PR_1(PU_j(k))$$

Obsérvese que si $j = 1$ entonces X es exactamente el valor de la clave escogida por B:

$$X = PR_1(PU_1(k)) = k$$

Y por el contrario si $j = 2$, entonces el descifrado dará como resultado un valor arbitrario:

$$X = PR_1(PU_2(k)) = ?$$

La probabilidad de que A se encuentre en una u otra solución es del 50%, porque A no tiene manera de saber la clave pública que escogió B (porque A no conoce la clave k que generó B).

A continuación A cifrará un mensaje "cruz" con el valor obtenido X, y transmitirá el resultado a B:

$$\text{A} \rightarrow \text{B: } E_X(cruz)$$

B realizará la operación de descifrado:

$$R = D_k(E_X(cruz))$$

En el caso de que j sea 1 (B escogió la primera clave pública) tendremos que X se corresponde con k y por tanto:

$$R = D_k(E_k(cruz)) = cruz$$

En caso contrario ($j = 2$) tendremos que:

$$R = D_k(E_X(cruz)) = ????$$

Y por convención supondremos que este resultado se corresponde con la "cara".

Para finalizar la ejecución del protocolo A y B intercambiarán la siguiente información:

$\text{A} \rightarrow \text{B: } PR_1, PR_2$
$\text{B} \rightarrow \text{A: } k$

para que ambos puedan comprobar que ninguno hizo trampa, es decir, que el 50% de las veces saldrá cara y el 50% de las veces cruz, sin que ninguna de las dos partes pueda saber (ni influir) en cuál será el resultado de la ejecución del protocolo.

Más allá del juego de "cara o cruz" lo que interesa del anterior protocolo es que una de las partes A, ha proporcionado a la otra uno de dos posibles valores

(cara o cruz) con la misma probabilidad, y sin que A sepa cuál de los dos le ha proporcionado. A esta característica se le denomina transferencia transcordada. Para explicar un protocolo de firma de contratos sin TTP, supondremos que disponemos de un protocolo similar, que etiquetaremos como:

$$\text{A} \rightarrow \text{B: } trans - transc(x,y)$$

que significa que A transmite a B uno de dos posibles valores (x o y), de manera equiprobable y sin que A sepa cuál de los dos ha transferido.

Protocolo de firma de contratos sin TTP

Explicamos aquí un protocolo de firma de contratos sin TTP, es decir, en la ejecución de este protocolo sólo intervendrán los dos firmantes del contrato.

El protocolo se inicia por parte de A generando de manera aleatoria $2N$ claves de criptografía simétrica (por ejemplo, de AES o DES), y las agrupa formando parejas:

$$\text{A: } (a_1,a_{N+1}),(a_2,a_{N+2}),\ldots,(a_N,a_{2N})$$

A continuación cifra un texto arbitrario S con las $2N$ claves que ha generado anteriormente, y envía los $2N$ criptogramas a B:

$$\text{A} \rightarrow \text{B: } C_i = E_{a_i}(S) \text{ (para } i = 1 \text{ hasta } 2N)$$

A acuerda con B que quedará vinculado a un contrato M si B puede obtener una o más de las parejas de claves que ha generado en el primer paso (y que en este momento solo ella conoce). Este paso queda fuera de la explicación detallada del protocolo, pero apuntamos que supondría enviar un mensaje firmado, indicando este extremo.

En este momento B procedería de manera análoga a los pasos que ha realizado A. En primer lugar generaría $2N$ claves de criptografía simétrica y las agruparía en parejas:

$$\text{B: } (b_1,b_{N+1}),(b_2,b_{N+2}),\ldots,(b_N,b_{2N})$$

A continuación también cifra el texto arbitrario S con las $2N$ claves que ha generado anteriormente, y envía los $2N$ criptogramas a A:

$$\text{B} \rightarrow \text{A: } D_i = E_{b_i}(S) \text{ (para } i = 1 \text{ hasta } 2N)$$

B también acuerda con A que quedará vinculado a un contrato *M* si A puede obtener una o más de las parejas de claves que ha generado (y que en este momento sólo B conoce).

En el siguiente paso es cuando necesitamos el protocolo de transferencia transcordada que apuntábamos anteriormente. A y B lo utilizarán para transmitir una de las dos claves de cada uno de los N pares de claves que han generado respectivamente. Empezaremos por A:

$$A \rightarrow B: trans - transc(a_i, a_{i+N}) \text{ (para todas las parejas de claves)}$$

Al finalizar este paso, A habrá transferido a B una de las dos claves de cada par, con dos características muy importantes para la seguridad del protocolo: A no sabe cuál de los dos claves de la pareja le ha proporcionado a B, y el hecho de que haya proporcionado una u otra (de cada par) es equiprobable.

B debe realizar el mismo paso con sus pares de claves:

$$B \rightarrow A: trans - transc(b_i, b_{i+N}) \text{ (para todas las parejas de claves)}$$

En este punto de la ejecución del protocolo A y B disponen de la mitad de los secretos de la otra parte. Esto no supone un compromiso para ninguno de los dos, pues han acordado que para dar por firmado el contrato deben disponer como mínimo de uno de los pares de las claves (por tanto, en este momento no disponen de la información suficiente).

A continuación A y B intercambiarán bit a bit todas las claves que generaron en la primera fase del protocolo. Este intercambio lo realizarán de manera intercalada, de manera que en todo momento ambos dispongan aproximadamente de la misma información de la otra parte. Obviamente, la parte que inicie el intercambio estará en desventaja, pero si el intercambio es bit a bit, esta desventaja será muy pequeña. Veamos cómo quedaría el protocolo:

FOR i = 1 TO L DO (donde L es la longitud de cada clave)
$A \rightarrow B$: el bit *i* de todas las claves a_i
$B \rightarrow A$: el bit *i* de todas las claves b_i

Si ninguna de las dos partes aborta la ejecución de este paso, al final del mismo A y B dispondrán de los *N* pares de claves de la otra parte. Ambas partes pueden verificar que los pares son correctos realizando el descifrado de los criptogramas que han intercambiado al inicio de la ejecución del protocolo. Era suficiente con un par de claves para darse por firmado el contrato, por tanto, el contrato quedará firmado.

Analicemos ahora qué sucede si una de las dos partes intenta hacer trampas. El primer intento de realizar un fraude podría producirse en el momento de

transferir una de las dos claves con la transferencia trascordada, es decir, intentar enviar una clave que no fue utilizada para cifrar el texto arbitrario *S*. De esta manera, la parte fraudulenta lo que intenta es obtener una par de claves de la otra parte, y dejar a la otra parte sin la posibilidad de obtener sus pares de claves. La transferencia trascordada no permite este ataque, pues está claro que el atacante no puede cambiar las dos claves. Una de las dos es transferida en el paso de transferencia trascordada, y por tanto, si las dos claves son "falsas" el otro extremo lo detectará rápidamente, pues esa clave no se corresponderá con ninguno de los criptogramas recibido en el primer paso del protocolo.

Por tanto, el atacante sólo puede cambiar una de las dos componentes de cada par. Pero tampoco podrá realizar este ataque, pues recuérdese que se transfiere una clave de cada par de manera transcordada, es decir, el emisor no sabe cuál de las dos componentes ha recibido el otro extremo. Si el emisor cambia una de las dos componentes aleatoriamente, en un par tiene el 50% de probabilidades de ser detectado por la otra parte (que se haya transferido la clave cambiada). Ahora se observa la importancia de que sean N pares de claves. Pues hemos visto que la probabilidad de no ser detectado en una transferencia es del 50% (que tenga la suerte de que el otro extremo reciba la clave correcta, la no cambiada). En un segundo par de claves, la probabilidad de no ser detectado también será del 50%, pero la probabilidad acumulada de no ser detectado es del 0,5*0,5 = 0,25, es decir, el 25%, y así sucesivamente, llegando a una probabilidad para N pares de:

$$\text{Prob\_no\_detección} = 2^{-N}$$

Para una valor relativamente pequeño de *N*, la probabilidad de no ser detectado puede hacerse despreciable. Como conclusión no es posible esta vía de ataque.

Una segunda posibilidad de hacer trampas que tienen A y B es intentar enviar bits incorrectos durante la fase de intercambio de las claves bit a bit. Veamos que tampoco es posible este ataque. En cada paso del intercambio bit a bit de las claves, A y B proporcionan un bit de cada clave de las $2N$ claves que han generado en el primer paso del protocolo. Pero en este punto el otro extremo dispone de una de las dos claves de cada par de manera segura (como acabamos de ver). Si A o B intentan enviar bits incorrectos de las dos claves de cada par, serán inmediatamente detectados, pues la otra parte dispone de una de las dos.

Si intentan enviar bits incorrectos de una de las dos claves de cada par, también serán detectados, pues no saben de cuál de las dos claves dispone la otra parte. Si una parte detecta que recibe bits incorrectos, abortará inmediatamente la ejecución del protocolo para no quedar en situación de desventaja (proporcionar un par a la otra parte y quedarse sin la posibilidad de disponer de un par de la otra parte). Por tanto, tampoco es posible este ataque.

Como conclusión, podemos afirmar que A y B no pueden realizar ninguno de los dos ataques planteados, sin asumir un muy elevado riesgo de ser detectado su intento de hacer trampas.

Podría concluirse que el protocolo presentado es suficientemente seguro, y sin necesidad de TTP, pero para ello hay que realizar la asunción de que las dos partes disponen de la misma potencia de cálculo. Para observar la necesidad de esta asunción, debemos plantearnos qué sucede si una de las dos partes aborta la ejecución del protocolo en la fase de intercambio de bits cuando ya se han intercambiado X bits de cada clave. Es cierto que cada parte dispone aproximadamente de la misma información que la otra (a lo sumo existe la diferencia de un bit más para B, si A es el primero que transmite sus bits).

Una vez abortada la ejecución del protocolo, la única posibilidad que les queda a las dos partes es lo que se conoce como un ataque por fuerza bruta, es decir, probar las combinaciones de bits posibles de los que no han sido proporcionados por la otra parte. Si han abortado la ejecución cuando sólo faltaba un bit, ambas partes podrán deducir el bit que falta sin mayor problema, pues solo deben realizar dos posibles pruebas (0 o 1). Pero si restan t bits, deberán realizar 2^t pruebas, lo que en función de t puede requerir una elevada potencia de cálculo para que el ataque pueda fructificar. Aquí es donde radica el problema del protocolo: lo que para una parte puede ser un problema trivial (dispone de la potencia de cálculo necesaria para realizar las 2^t pruebas en un tiempo razonable) para la otra parte puede ser un problema irresoluble (puede necesitar un tiempo y recursos de los que no dispone). Suponer que las partes que intervienen en la firma de un contrato disponen de la misma potencia de cálculo es poco realista y peligroso desde el punto de vista de seguridad (compárese la potencia de cálculo de un particular frente a la potencia de cálculo de Google). Por este motivo, no suelen recomendarse las soluciones sin TTP, a pesar de la aparente ventaja de no necesitar de la intervención de actores diferentes de los firmantes del contrato.

Las soluciones sin TTP basadas en el intercambio gradual de secretos suelen se descartadas porque para garantizar su seguridad realizan la asunción de que las partes disponen de la misma potencia de cálculo, asunción que es poco realista en la práctica.

3. Soluciones para la firma electrónica de contratos multiparte

Hasta el momento hemos presentado soluciones para el caso de que sean dos las partes contratantes, pero también puede darse el caso de que sean más de dos usuarios los que deban firmar un mismo contrato. Hablamos así de protocolos multiparte para la firma electrónica de contratos.

Solución para la firma electrónica de contratos multiparte con TTP *in-line*

Como en el caso de los contratos entre dos partes empezaremos con la solución más sencilla posible. Se trata de una solución con TTP *in-line*, en la que todas las partes intervinientes (C_1 a C_N) envían su copia firmada a la TTP:

$$C_i \rightarrow \text{T: } Sign_i(M)$$

Una vez que la TTP dispone de las copias de todas las partes, verificará su corrección y en caso de que sean todas correctas, deberá retransmitir las $N - 1$ copias de las otras partes a cada participante:

$$\text{T} \rightarrow C_i\text{: } Sign_j(M) \text{ (para } j = 1 \text{ hasta } N, \, j \text{ distinto de } i)$$

Con esta sencilla solución conseguimos la propiedad más importante de la firma electrónica de contratos: la equidad. Pero esta solución comparte los mismos defectos que la equivalente para dos partes, especialmente, el posible coste de una TTP que debe intervenir en todas las ejecuciones del protocolo, y con un coste computacional considerable.

Solución para la firma electrónica de contratos multiparte con TTP *off-line* síncrona

Seguimos considerando que las soluciones optimistas son preferibles, pues reducir la intervención de la TTP debe significar reducir el riesgo de que se convierta en un cuello de botella, o como mínimo reducir los costes asociados. Por ello a continuación exponemos una solución optimista síncrona, lo que significa que se establece un plazo, una fecha límite para llevar a cabo la firma del contrato. De hecho, veremos que la fecha límite, más que con la firma del contrato directamente, tiene que ver con la intervención de la TTP.

Ya hemos visto que un protocolo optimista debe contar con un subprotocolo en el que sólo intervienen los firmantes (subprotocolo de intercambio) y un subprotocolo (o más de uno) en el que debe intervenir la TTP.

Empezaremos explicando el subprotocolo de intercambio:

$$C_i \rightarrow C_j\text{: } Sign_i(M) \text{ (para } i,j = 1 \text{ hasta } N,\ i \text{ distinto de } j)$$

En un primer paso cada firmante debe enviar su ejemplar firmado a todas las demás partes. A continuación cada firmante debe enviar un acuse de recibo a los demás firmantes, para confirmar que ha recibido la copia firmada de los otros:

$$C_i \rightarrow C_j\text{: } Sign_i(Sign_j(M)) \text{ (para } i,j = 1 \text{ hasta } N,\ i \text{ distinto de } j)$$

Los acuses de recibo son necesarios como elemento de prueba para demostrar que el contrato está firmado. Por tanto, no es suficiente con recibir las copias firmadas, sino que también son necesarios los acuses de recibo.

Como en el caso de la firma entre dos partes, ahora debe proporcionarse un subprotocolo para el caso de que alguna de las partes no cumpla con lo que está previsto en el subprotocolo de intercambio. Establecemos que las partes tienen un tiempo t para contactar con la TTP, tiempo tras el cual la TTP no aceptará peticiones de resolución con relación a ese contrato, y lo único que hará es notificar la resolución a la que llegó antes de ese periodo, si procede.

Si una parte ha enviado sus acuses de recibo, y no está recibiendo los de una o más de las contrapartes, debe enviar a la TTP todas las pruebas acumuladas del intercambio, que como mínimo contendrán las firmas sobre el contrato de los demás, pues en caso contrario ese usuario no debería haber enviado sus acuses de recibo:

$$C_i \rightarrow \text{T: pruebas acumuladas}$$

Las pruebas aportadas por C_i demuestran que todas las partes se han comprometido a firmar el contrato, y por tanto que saben que tienen iniciada la ejecución del protocolo. Siendo así, la TTP debe proporcionar un acuse de recibo equivalente a los que deberían haber enviado las otras partes contratantes a C_i:

$$\text{T} \rightarrow C_i\text{: } Sign_T(\text{acuse de recibo equivalente})$$

Si alguna de las otras partes tampoco ha recibido todos los acuses de recibo, también sabe que debe contactar con la TTP antes de que venza t, bajo el riesgo de quedarse sin pruebas de la firma del contrato.

Es sencillo ver que el protocolo propuesto cumple la propiedad de equidad.

Solución para la firma electrónica de contratos multiparte con TTP *off-line* asíncrona

De la misma manera que hemos presentado una solución asíncrona para el caso de dos firmantes, también existen propuestas de solución para el caso de que sean múltiples los firmantes. Las soluciones para el caso de múltiples firmantes suelen ser complejas y por ello, en lugar de proporcionar una solución genérica para el caso de N firmantes, explicaremos una solución para un caso particular de 3 firmantes.

El protocolo consta de tres subprotocolos como el caso de dos partes: un subprotocolo de intercambio, uno de cancelación y uno de finalización. Empezaremos describiendo el subprotocolo de intercambio. El intercambio de información entre los actores podría realizarse de múltiples maneras. Uno de ellos podría controlar el intercambio, adoptando una topología en estrella. También podría optarse por una solución similar a la del caso síncrono, es decir, que todos envían la información a todos. Aquí optamos por utilizar una topología en anillo, en la que cada parte asume una posición en el anillo, recibe una información de su predecesor, y transmite parte de esa información y la que ella genera al usuario que va después.

En este escenario aparecerá un actor nuevo C (Charles). Veamos cómo es el subprotocolo de intercambio en la topología en anillo:

A → B: M,$Sign_A(M,1)$
B → C: M,$Sign_A(M,1)$,$Sign_B(M,1)$
C → A: $Sign_B(M,1)$,$Sign_C(M,1)$
A → B: $Sign_C(M,1)$,$Sign_A(M,2)$
B → C: $Sign_A(M,2)$,$Sign_B(M,2)$
C → A: $Sign_B(M,2)$,$Sign_C(M,2)$
A → B: $Sign_C(M,2)$,$Sign_A(M,3)$
B → C: $Sign_A(M,3)$,$Sign_B(M,3)$

El subprotocolo consta de tres rondas, y las pruebas que se intercambian en cada transmisión incorporan un índice que indica a qué ronda corresponden. Veremos que esto será necesario para que la TTP pueda resolver las peticiones de resolución, en caso de que sea necesaria su intervención. Las pruebas de que el contrato está firmado son la acumulación de las firmas realizadas y recibidas de los otros actores.

Puede demostrarse que en un escenario de firma electrónica de contratos asíncrona y con TTP *off-line* son necesarias más de $N - 1$ rondas si tenemos N participantes, y de aquí la complejidad de explicar la solución para el caso genérico.

Retomando la explicación del caso de tres usuarios, si ninguna de las partes aborta la ejecución del subprotocolo de intercambio, y si no se presentan

problemas de comunicaciones, cada parte dispondrá de las pruebas necesarias para demostrar que el contrato está firmado, y sin necesidad de que intervenga la TTP.

Pero como en casos anteriores, ahora debemos plantearnos qué sucede si surgen problemas de comunicaciones, o alguna de las partes intenta hacer trampas. Obviamente, los contratantes honestos deben contactar con la TTP para recuperar la equidad del intercambio. Pero, ¿cuáles son las reglas que debe seguir la TTP para garantizar la equidad de todas las partes honestas?

Construiremos los subprotocolos de cancelación y finalización paso a paso (simplificando o aunando casos siempre que sea posible), considerando las posibles situaciones que pueden darse. Desde el momento en que A, B y C tengan pruebas del compromiso de las otras dos partes, podrán acudir a la TTP para que esta pueda dar por finalizado el intercambio. Es decir, en todos los siguientes casos:

A → T: $Sign_B(M,1)$,$Sign_C(M,1)$
B → T: $Sign_A(M,1)$,$Sign_C(M,1)$
C → T: $Sign_A(M,1)$,$Sign_B(M,1)$
B → T: $Sign_A(M,1)$,$Sign_C(M,1)$,$Sign_A(M,2)$,$Sign_C(M,2)$
C → T: $Sign_A(M,1)$,$Sign_B(M,1)$,$Sign_A(M,2)$,$Sign_B(M,2)$

la TTP les devolverá un acuse de recibo, indicando que se da por firmado el contrato M:

$$\text{T} \rightarrow \text{A/B/C: } Sign_T(M)$$

Esta decisión de la TTP es irrevocable, es decir, una vez que la TTP haya decidido que el contrato se da por firmado, no existe ninguna petición posterior a la TTP que pueda hacer variar esa decisión. Esta regla de la TTP se basa en el hecho de que las pruebas aportadas por las partes demuestran que todos habían manifestado su intención de firmar el contrato, y que por tanto, si no habían recibido las pruebas necesarias de los demás sabían que debían contactar con la TTP, que les aportará la prueba de que el contrato se da por firmado.

Quedan por resolver las posibles peticiones de A y B, en el caso de que afirmen que han enviado los mensajes de la primera ronda y no reciben las pruebas que esperan a cambio. Al tratarse de un protocolo asíncrono, pero en el que no queremos que las partes implicadas deban esperar un tiempo indefinido para saber si el contrato será firmado o no, debemos permitir que contacten con la TTP:

A → T: $Sign_A(M,1)$
B → T: $Sign_A(M,1)$,$Sign_B(M,1)$

Con la anterior información, y si nadie había contactado previamente con la TTP para finalizar el intercambio (decisión que hemos dicho que es irrevocable) la TTP no puede hacer otra cosa que cancelar el intercambio:

$$T \rightarrow A/B: Sign_T(firma - de - M - cancelada)$$

No tiene sentido permitir que C pueda cancelar el intercambio, porque cuando C puede contactar con la TTP ya tiene suficiente información para que la TTP pueda decidir que se debe dar por firmado el contrato (véase el caso anterior).

A diferencia de la decisión de la TTP de que el contrato está firmado, la de cancelación sí puede ser revocada por la TTP. Esto es debido a que podría ser que A o B estuvieran haciendo trampas y esto se pudiera detectar a posteriori (con la información aportada por otras partes). Para ello debemos analizar las posibles situciones de contacto con la TTP que pueden darse tras una cancelación de A o B. Realizaremos el análisis para el caso en que A es la primera parte que cancela (quedando como ejercicio el caso en que sea B el primero que cancela):

- A cancela, B cancela, C finaliza con la información de la primera ronda o no contacta con la TTP

  En este caso, y en todos los que siguen, ante la petición de A, la TTP inicialmente da el intercambio por cancelado. La petición de cancelar de B es compatible con la anterior de A, y por tanto, la TTP sigue manteniendo el estado del intercambio como cancelado. Finalmente, la petición de C también es compatible con lo anterior (puede ser que C haya enviado su información de la primera ronda, pero por problemas con el canal de comunicaciones esa información no llegó a A), y por tanto la TTP mantiene el estado del intercambio a cancelado.

- A cancela, B cancela, C finaliza con la información de la segunda ronda

  En este caso, la petición de C demuestra a la TTP que A y B habían recibido la información que C había enviado, porque de lo contrario A y B no hubiesen proporcionado a C la información de la segunda ronda. Siendo así la TTP concluye que A y B mintieron y cambia el estado del intercambio a finalizado y proporciona la firma del contrato a C:

  $$T \rightarrow C: Sign_T(M)$$

- A cancela, B finaliza con la información de la segunda ronda, independiente de lo que haga C

  Como en el caso anterior, el hecho de que B aporte pruebas de la segunda ronda demuestra que A hizo trampas (disponía de más información de la

que aportó a la TTP, pues de lo contrario no hubiese enviado las pruebas de la segunda ronda a B). La TTP dará el contrato por firmado:

$$\mathrm{T} \rightarrow \mathrm{B}: \mathit{Sign}_T(M)$$

- A cancela, C finaliza con la información de la primera ronda, B cancela o no contacta con la TTP

  Como en el primer caso analizado, el hecho de que C aporte las pruebas de la primera ronda es compatible con la petición de A, y por tanto la TTP mantendrá el estado del intercambio como cancelado.

- A cancela, C finaliza con la información de la segunda ronda, independiente de lo que haga B

  Las pruebas que aporta C de la segunda ronda demuestran que A hizo trampas (disponía de más información de la que aportó a la TTP, pues de lo contrario no hubiese enviado las pruebas de la segunda ronda a B, y éste a su vez a C). La TTP dará el contrato por firmado.

- A cancela, C finaliza con la información de la primera ronda, B contacta con la información de la segunda ronda

  Como ya hemos dicho, la información aportada por C es compatible con la petición inicial de A, y por tanto la TTP mantiene el estado de cancelado, y le envía a C la cancelación:

  $$\mathrm{T} \rightarrow \mathrm{C}: \mathit{Sign}_T(\mathit{firma-de-M-cancelada})$$

  A continuación B contacta con al TTP y le aporta una información que demuestra que A hizo trampas (tenía más información de la que aportó), pero la TTP no puede revocar la decisión tomada, porque en caso contrario podría estar perjudicando a C. Por tanto, a pesar de que A había hecho trampas, como puede ser que C no hubiera hecho trampas, la TTP mantiene el estado de cancelado y envía tal decisión a B.

El último caso (el encadenamiento de cancelaciones) es el que justifica la necesidad de tres rondas. En ningún caso debe permitirse que una parte que sea honesta pierda la equidad del intercambio por una decisión incorrecta de la TTP. Puede probarse fácilmente que con dos rondas la intervención de la TTP no permitiría resolver adecuadamente todas las posibles situaciones.

Para finalizar, podemos concluir que el protocolo presentado garantiza la equidad para las partes honestas, siendo del tipo optimista y asíncrono.

Ejercicios de autoevaluación

1. Convertid la solución con TTP *in-line* para la firma electrónica de contratos entre dos partes, en una solución con TTP *on-line*.

2. Verificad si en la solución para la firma electrónica de contratos (entre dos partes) con TTP *off-line* de cuatro pasos, efectivamente el comportamiento de la TTP es verificable.

3. Proporcionad un protocolo para la firma electrónica de contratos con TTP *off-line* en el caso en que sean cuatro las partes que han de firmar el mismo contrato.