(19) OFICINA ESPAÑOLA DE PATENTES Y MARCAS

ESPAÑA

(11) Número de publicación: **2 409 458**

(21) Número de solicitud: 201130574

(51) Int. CI.:

H04L 9/00 (2006.01)

(12) SOLICITUD DE PATENTE A2

(22) Fecha de presentación:

12.04.2011

(43) Fecha de publicación de la solicitud:

26.06.2013

(71) Solicitantes:

TELEFÓNICA, S.A. (100.0%)
C/ GRAN VÍA, 28
28013 MADRID ES

(72) Inventor/es:

MONTOJA VITINI, Fausto ;
ORÚE LOPEZ, Amalia Beatriz ;
GUERRA ESTEVEZ, Alberto ;
HERNANDEZ ENCINAS, Luis;
MARTIN MUÑOZ, Agustin y
SOTO RODRIGUEZ, Mercedes

(74) Agente/Representante:

CARPINTERO LÓPEZ, Mario

(54) Título: **MÉTODO Y SISTEMA PARA MEJORAR LA SINCRONIZACIÓN DE CIFRADOS DE FLUJO**

(57) Resumen:

Método y sistema para mejorar la sincronización de cifrados de flujo.

Método y sistema para mejorar la encriptación de datos por cifrado de flujo, en el que los datos que van a encriptarse, denominados texto simple, se combinan con una secuencia aleatoria de dígitos, denominada flujo de clave, para obtener el texto simple encriptado denominado texto cifrado. La invención propone un nuevo tipo de sincronismo de cifrado de flujo, en el que la acción de sincronización se produce a intervalos aleatorios con una planificación temporal impredecible.

ES 2 409 458 A2

DESCRIPCIÓN

Método y sistema para mejorar la sincronización de cifrados de flujo

CAMPO TÉCNICO

La presente invención se refiere, en general, a la encriptación de datos y más en particular a un método y a un sistema para mejorar la sincronización y, por tanto, el comportamiento de procedimientos de encriptación por cifrado de flujo.

DESCRIPCIÓN DE LA TÉCNICA ANTERIOR

La encriptación es el proceso de transformar información (denominada texto simple, texto sin codificar o texto en claro) usando un algoritmo (denominado cifrado) para hacer que sea ilegible para cualquiera excepto para aquéllos que tienen un conocimiento especial, normalmente denominado clave. El resultado del proceso es información encriptada (en criptografía denominada texto cifrado). En muchos contextos, el término encriptación también se refiere implícitamente al proceso inverso, desencriptación, hacer legible de nuevo la información encriptada (es decir, desencriptarla).

La encriptación se usa en la actualidad habitualmente en la protección de información en muchas clases de sistemas. La encriptación puede usarse para proteger datos “en reposo”, tales como archivos en ordenadores y dispositivos de almacenamiento. En los últimos años, numerosos informes de datos confidenciales, tales como registros personales de clientes, han quedado expuestos debido a pérdida o robo de ordenadores portátiles o unidades de copia de seguridad. La encriptación de tales archivos en reposo ayuda a protegerlos si fallan las medidas de seguridad físicas. Los sistemas de gestión de derechos digitales que impiden el uso o la reproducción no autorizados de material sujeto a *copyright* y protegen el software frente a ingeniería inversa son otro ejemplo algo diferente de uso de la encriptación de datos en reposo. La encriptación también se usa para proteger datos en tránsito, por ejemplo datos que están transfiriéndose a través de redes (por ejemplo, Internet, comercio electrónico), teléfonos móviles, micrófonos inalámbricos, sistemas de intercomunicación inalámbricos, dispositivos Bluetooth y cajeros automáticos. Se han interceptado numerosos informes de datos en tránsito en los últimos años. La encriptación de datos en tránsito también ayuda a protegerlos ya que a menudo es difícil proteger físicamente todos los accesos a redes.

Un cifrado de flujo es un tipo de algoritmo de encriptación simétrico. La encriptación se logra combinando el texto simple (el mensaje que va a encriptarse) con un flujo de clave, normalmente con la operación de O exclusivo (XOR), aunque también son posibles otras operaciones. La combinación se realiza dígito a dígito; normalmente los dígitos consisten en uno o varios bits. La transformación de bits de texto simple sucesivos varía durante la encriptación, se encriptan bits de texto simple iguales en diferentes bits de texto cifrado dependiendo de su posición en el mensaje.

El flujo de clave consiste en una secuencia aleatoria de bits, que puede generarse por adelantado o en el momento de uso. Esta secuencia de bits puede representar cualquier dato, cuya confidencialidad deba protegerse, tanto para su almacenamiento como para su transmisión: texto, imágenes, programas, audio y vídeo. Para garantizar la inquebrantabilidad del cifrado, la longitud del flujo de clave debe ser igual a la longitud del texto simple, y cada flujo de clave no debe reutilizarse para más de una operación de encriptación. Puesto que todo el flujo de clave es aleatorio, incluso un enemigo con infinitos recursos informáticos sólo puede adivinar el texto simple si ve el texto cifrado. Se dice que un cifrado de este tipo ofrece un secreto perfecto.

Los precedentes históricos del cifrado de flujo son el cifrado César, cuyo flujo de clave sólo tenía un dígito que consistía en un carácter alfabético, y el cifrado de Blaise de Vigenere, cuya clave tenía un número limitado de dígitos y se volvía a reproducir tantas veces como fuese necesario para alcanzar la longitud del texto simple. El primer cifrado de flujo moderno fue el cifrado Vernam, también conocido como libreta de un solo uso, que es el único cifrado inquebrantable que puede demostrarse matemáticamente. La libreta de un solo uso se usó en tiempo de guerra en canales diplomáticos que requerían una seguridad excepcionalmente alta. El hecho de que la clave secreta (que puede usarse sólo una vez) sea tan larga como el mensaje introduce graves problemas de gestión de claves. Aunque es perfectamente segura, la libreta de un solo uso es en general poco práctica. Los cifrados de flujo actuales se desarrollaron como una aproximación a la acción de la libreta de un solo uso. Aunque los cifrados de flujo contemporáneos no pueden proporcionar la seguridad teórica satisfactoria de la libreta de un solo uso, son al menos prácticos.

La generación del flujo de clave se logra mediante un generador de números pseudoaleatorios (PRNG; del inglés pseudo-random numbers generator) o mediante determinados modos de operación de cifrado por bloques, que los transforman de manera eficaz en un generador de flujos de clave; de esta manera, puede usarse cualquier cifrado por bloques como cifrado de flujo: como en los modos de operación TDEA (algoritmo triple de encriptación de datos) o AES (sistema de encriptación avanzada) en OFB (realimentación abierta) o CRT (contador). Sin embargo, los cifrados de flujo con un diseño dedicado son normalmente mucho más rápidos. Los PRNG o cifrados por bloques deben tener una clave *K* que pueda determinar la función de siguiente estado y la función de salida así como el estado inicial. El problema básico del diseño de generador de flujos de clave en el contexto de máquinas de estados finitos es hallar las funciones de siguiente estado y las funciones de salida que garanticen que se produce un flujo de clave que satisface

los requisitos básicos de las propiedades de periodo grande, gran complejidad lineal y distribución uniforme. La determinación de estos criterios de rendimiento básicos implica, naturalmente, la analizabilidad del generador de flujos de clave. La construcción de generadores de claves de funcionamiento seguro implica necesariamente la introducción de transformaciones no lineales, lo que complica enormemente el análisis indispensable. Existen métodos implícitos y explícitos para introducir efectos no lineales.

La generación del flujo de clave puede se independiente del texto simple y el texto cifrado, produciendo lo que se denomina un cifrado de flujo síncrono, o puede depender de los datos y su encriptación, en cuyo caso el cifrado de flujo se dice que es de autosincronización. El cifrado Vernam era del primer tipo.

Un generador de flujos de clave puede describirse mediante una 6-tupla (*S, R, f, g, K*1*, i*), de manera más precisa:

S, conjunto de estados internos;

R, alfabeto de salida;

f: *f* (*S, K*1)→*S,* la función de transición de estados;

g: g(*S, K*1)→*R,* la función de salida;

*K*1, la clave de las funciones *f* y *g*;

i, el valor inicial (*seed*) que determina el estado inicial.

El número de estados internos *S* es mucho mayor que el alfabeto de salida *R*; cada estado es una función *f* de la clave *K*1 y el estado previo (la clave K1 suele ser una secuencia de dígitos); la salida en cada momento es una función *g* de la clave *K*1 y el estado correspondiente; el estado inicial está determinado por la constante *i.* Cada vez que cambia que el estado interno se genera un nuevo dígito de salida, que es una función del nuevo estado y de la clave *K*1; el momento de cambio del estado interno se controla normalmente mediante un reloj interno o un mecanismo de sincronismo entre un transmisor y un receptor.

Existen dos tipos principalmente de cifrado de flujo, los cifrados de flujo síncronos y los cifrados de flujo de autosincronización.

En un cifrado de flujo síncrono se genera un flujo de dígitos pseudoaleatorios independientemente de los mensajes de texto simple y texto cifrado, y luego se combina con el texto simple (para la encriptación) o con el texto cifrado (para la desencriptación). En la forma más común se usan dígitos binarios (bits) y el flujo de clave se combina con el texto simple usando O exclusivo (XOR) (también pueden usarse otras funciones lógicas). Esto se denomina cifrado de flujo aditivo binario.

En un cifrado de flujo síncrono, el emisor y el receptor deben estar exactamente en sintonía para que la desencriptación sea satisfactoria. Si se añaden o se eliminan dígitos para el mensaje durante la transmisión, se pierde la sincronización. Para restaurar la sincronización, pueden probarse sistemáticamente diferentes desviaciones para obtener la desencriptación correcta. Otro planteamiento es etiquetar el texto cifrado con marcadores en puntos regulares en la salida. Sin embargo, si se corrompe un bit en la transmisión, en lugar de añadirse o perderse, sólo se ve afectado un único bit en el texto simple y el error no se propaga a otras partes del mensaje. Esta propiedad es útil cuando la tasa de errores de transmisión es alta; sin embargo, hace que sea menos probable que se detecte el error sin mecanismos adicionales.

En cifrados de flujo síncronos, el siguiente estado depende sólo del estado previo y no de la entrada, de modo que la sucesión de estados es independiente de la secuencia de caracteres recibidos. Por consiguiente, la transformación de cifrado no tiene memoria, aunque varía con el tiempo. No obstante, el dispositivo propiamente dicho sí tiene memoria, necesita la memoria interna para generar la secuencia de estados necesaria. Por tanto, es natural en un cifrado de flujo síncrono separar la transformación de cifrado del proceso de generación del parámetro que varía con el tiempo, que controla la transformación de cifrado.

La figura 1 ilustra el diagrama de bloques completo de un sistema de cifrado y descifrado de flujo síncrono. El texto simple (3) se encripta para dar el texto cifrado (5), mediante la suma módulo 2 (XOR) (4) de los bits de texto simple con los bits de flujo de clave generados por el PRNG (2); este PRNG genera un flujo de clave que es una función complicada de la clave, que se almacena en el registro (1). Si tanto el texto simple como el flujo de clave se componen de bits independientes, la operación XOR se lleva a cabo bit a bit; si tanto el texto simple como el flujo de clave se componen de dígitos formados por bloques de varios bits, la operación XOR se realiza a nivel de bits, es decir bit a bit. Por ejemplo, si el dígito de texto simple *pd* está formado por los 4 bits *pd* = (*pb*1, *pb*2, *pb*3, pb4) y el dígito de clave *kd* está formado por los 4 bits *kd* = (*kb*1, *kb*2, *kb*3, kd4), la operación XOR será: *pd* XOR *kd* = (*pb*1 XOR *kb*1, *pb*2 XOR *kb*2, *pb*3 XOR *kb*3, *pb*4 XOR *kb*4).

El texto cifrado se transmite al extremo receptor, en el que se desencripta para dar el texto simple recuperado

(10), mediante la resta módulo 2 (9) de los bits de flujo de clave, generados por el PRNG (7), a partir de los bits de texto cifrado (8), las operaciones de suma y resta módulo 2 son idénticas, de ahí que ambas se implementen mediante un mecanismo único: la combinación XOR de bits; el PRNG (7) genera un flujo de clave que es idéntico al generado por el PRNG (2), en función de la clave (6), cuya réplica exacta se almacena en el registro (1).

En los cifrados de flujo de autosincronización (también conocidos como cifrados de flujo asíncronos y también como autoclave de texto cifrado), el flujo de clave no es independiente de los mensajes de texto simple y texto cifrado debido a los *N* dígitos de texto cifrado previos para calcular el flujo de clave. La idea de la autosincronización se conoce desde 1946, y tiene la ventaja de que el receptor se sincronizará automáticamente con el generador de flujos de clave tras recibir *N* dígitos de texto cifrado, facilitando la recuperación del texto simple, si se quitan o añaden dígitos al flujo de mensaje. Los errores de un solo bit tienen un efecto limitado, afectando sólo a hasta *N* dígitos de texto simple.

Los cifrados de flujo de autosincronización tienen la capacidad de reanudar la desencriptación correcta si el flujo de clave generado por la unidad de desencriptación se desincroniza con respecto al flujo de clave de encriptación. Supóngase que la encriptación de un bit depende de *k* bits de texto cifrado previos. El sistema demuestra una propagación de errores limitada; si un bit se recibe incorrectamente, entonces la desencriptación de los siguientes *k* bits puede ser incorrecta. Sin embargo, adicionalmente el sistema puede volver a sincronizarse y producir una desencriptación correcta tras haberse recibido correctamente *k* bits. Esto hace que tales cifrados sean adecuados para aplicaciones en las que es difícil mantener la sincronización. A diferencia de los cifrados de flujo síncronos, en los cifrados de flujo de autosincronización, la transformación de descifrado tiene memoria finita con respecto a la influencia de caracteres anteriores de modo que un carácter de texto cifrado erróneo o perdido produce sólo un número fijo de errores en el texto simple descifrado, tras lo cual se produce de nuevo texto simple correcto. Para estos cifrados de flujo, la función que define el siguiente estado del sistema criptográfico toma como entrada algunos de los dígitos previos del texto cifrado generado. El ejemplo más común de esto lo proporciona un cierto cifrado por bloques en lo que se denomina el modo de realimentación de cifrado (CFB).

La figura 2 ilustra el diagrama de bloques completo de un sistema de cifrado y descifrado de flujo de autosincronización. El texto simple (3) se encripta para dar el texto cifrado (5), mediante XOR (4) de los bits de texto simple con los bits de flujo de clave generados por el PRNG (2); este PRNG genera un flujo de clave que es una función complicada de la clave (1), y de los últimos *N* dígitos del texto cifrado, almacenados en el registro (11). Si tanto el texto simple como el flujo de clave se componen de bits independientes, la operación XOR se lleva a cabo bit a bit; pero si tanto el texto simple como el flujo de clave se componen de dígitos formados por bloques de varios bits, la operación XOR se realiza a nivel de bits. El texto cifrado se transmite al extremo receptor, en el que se desencripta para dar el texto simple recuperado (10), mediante la resta módulo 2 (9) de los bits de flujo de clave, generados por el PRNG (7), a partir de los bits de texto cifrado (8); el PRNG (7) genera un flujo de clave que es idéntico al generado por el PRNG (2), en función de la clave (6) y de los últimos *N* dígitos del texto cifrado recibido, almacenados en el registro (12).

Los principales problemas de los cifrados de flujo existentes son los siguientes:

Los cifrados de flujo síncronos requieren un sincronismo perfecto entre el dispositivo de encriptación y el de desencriptación. Cuando se pierden algunos bits o se añaden intencionadamente durante la transmisión, entonces el flujo de clave producido en el sitio receptor se combinará con un cierto retraso o adelanto con respecto al texto cifrado provocando una nueva encriptación efectiva del texto cifrado una segunda vez. Restablecer el sincronismo en el receptor implica normalmente buscar todas las posibles desviaciones que pudiera haber experimentado el flujo de clave, o notificar al emisor que debería reinicializarse el sistema criptográfico. En ambos casos, puede perderse gran parte de los datos transmitidos. Esta dificultad de configurar de nuevo el sincronismo se considera que es la principal desventaja de los cifrados de flujo síncronos.

En la mayoría de los cifrados de flujo, si se pierde un bit de texto cifrado, o se inserta de algún modo un bit extra, se perderá la sincronización y los datos descifrados posteriores serán incorrectos. Sin embargo, también puede producirse una recepción ininteligible desde la posición del error hasta el final del mensaje a partir de un error de valor de texto cifrado cuando se usa un cifrado de flujo síncrono que presenta difusión de datos hacia delante.

Por otro lado, los cifrados de flujo de autosincronización protegen frente a la búsqueda de texto cifrado, pero debido a su propiedad de autosincronización no protegen (ó sólo ligeramente) frente a inyecciones, eliminaciones y nueva reproducción del texto cifrado. En general, los cifrados de flujo de autosincronización ofrecen un menor nivel de seguridad puesto que el criptoanalista conoce los argumentos y los valores de función del proceso de generación de los flujos de clave. La desventaja real de los cifrados de flujo de autosincronización es, sin embargo, su limitada analizabilidad debido a la dependencia del flujo de clave con respecto al flujo de mensaje.

Los cifrados de flujo de autosincronización tienen una cierta propagación de errores limitada que puede verse o no como una ventaja. Así es, cualquier cambio realizado por un atacante al texto cifrado tendrá consecuencias adicionales en otras partes del texto simple desencriptado.

Existen dos inconvenientes principales de los cifrados de flujo de autosincronización: 1) los cifrados de flujo de autosincronización también son vulnerables a un ataque de reproducción. En primer lugar, el atacante graba algunos bits de texto cifrado. Luego, en un momento posterior, sustituye esta grabación en el tráfico actual. Tras determinada

información inservible mientras se resincroniza el extremo receptor, el texto cifrado antiguo se desencriptará de manera normal. El extremo receptor no tiene forma de saber que no se trata de datos actuales, sino de datos antiguos que están volviéndose a reproducir. A menos que se usen sellos de fecha y hora, el atacante puede convencer a un banco para realice un ingreso en su cuenta una y otra vez, volviendo a reproducir el mismo mensaje (suponiendo que la clave no ha cambiado, por supuesto). Otra debilidad en este tipo de esquema podría aprovecharse en los casos de resincronización muy frecuente. 2) A un oponente que desee interceptar y desencriptar un mensaje cifrado le puede resultar más fácil averiguar el siguiente estado del PRNG puesto que su entrada se toma parcialmente del texto cifrado.

Sin embargo, los cifrados de flujo de autosincronización, cuando se realizan con funciones de flujo de clave no lineales, tienen la ventaja de combinar secreto con facilidad de sincronización. Son resistentes frente a desplazamientos de bits (inserción o eliminación de dígitos). Los errores de un solo bit en el texto cifrado dan como resultado ráfagas de errores en el texto simple, lo que es una desventaja para las aplicaciones de comunicación. Por otro lado, esto impide que intrusos activos realicen cambios selectivos de dígitos de texto simple individuales a través del texto cifrado y proporciona por tanto un nivel básico de autenticidad de mensaje; porque cualquier cambio de un dígito de texto cifrado no sólo inducirá el cambio del dígito de texto simple recuperado correspondiente (tras una decodificación legítima), sino que provocará una ráfaga de errores de los dígitos subsiguientes, revelando al receptor legítimo que el mensaje decodificado no es fiable porque un enemigo o simplemente un error de transmisión fortuito lo ha corrompido.

SUMARIO DE LA INVENCIÓN

La presente invención usa un nuevo método y sistema que reducirá o eliminará las deficiencias de las herramientas actuales.

El objeto de la presente invención es proponer un nuevo tipo de cifrados de flujo. La acción de sincronización se produce a intervalos aleatorios, en una planificación temporal impredecible; siendo imposibles de detectar por un observador no autorizado que no posea la clave. La invención proporciona, ventajosamente, una manera de mantener al emisor y al receptor de una transmisión encriptada exactamente en sintonía, evitando la pérdida de información cuando se produce una alteración del flujo de información entre los mismos, o bien por ruido o bien por la perturbación intencionada de un enemigo.

La aparición de la acción de sincronismo está inducida por la aparición espontánea de una sucesión de varios dígitos predeterminados en la secuencia de texto cifrado; cuando se observa esta sucesión de dígitos se realiza la sincronización, cambiando el estado del PRGN que generó el flujo de clave, siendo el nuevo estado una función no lineal de un conjunto de dígitos transmitidos previamente. Además, el esquema puede aumentar la seguridad de la transmisión encriptada aumentando el tamaño de la clave de sistema.

En un primer aspecto, se presenta un método para mejorar la encriptación de datos por cifrado de flujo, en el que los datos que van a encriptarse, denominados texto simple, se combinan con una secuencia aleatoria de dígitos, denominada flujo de clave, para obtener el texto simple encriptado denominado texto cifrado, generándose el flujo de clave mediante un generador de números pseudoaleatorios, PRNG, en función de una determinada secuencia de clave y el estado actual del generador, siendo los dígitos bloques de s bits, siendo s un parámetro de diseño, comprendiendo el método las siguientes etapas:

a) determinar el estado inicial del generador de números pseudoaleatorios y poner al estado cero, el contenido de una primera y una segunda línea de retardo (13a, 14a) del texto cifrado y el contenido de un primer y un segundo registro

b) obtener la primera clave, K1, que se usará como la secuencia de clave del PRNG

c) obtener una segunda clave, K2, que será una secuencia de l dígitos, siendo l un parámetro de diseño

d) retardar en la primera línea de retardo la secuencia actual del texto cifrado k dígitos y almacenar en el primer registro los últimos l dígitos del texto cifrado retardado, siendo k un parámetro de diseño, siendo $k \geq 0$.

e) comparar K2 con la secuencia de l bits del texto cifrado retardado almacenado en la etapa d)

f) si ambas secuencias no coinciden, generar el siguiente estado del PRNG en función de la primera clave y el estado previo y avanzar a la etapa j)

d) si ambas secuencias coinciden, retardar en la segunda línea de retardo el texto cifrado j dígitos y almacenar en el segundo registro los últimos m dígitos del texto cifrado retardado, siendo j y m parámetros de diseño, siendo $j \geq 0$

h) aplicar una función de transferencia a la secuencia de m dígitos almacenados en la etapa g) para obtener una secuencia transformada

i) generar una orden de carga al PRNG para cambiar el estado del PRNG a un valor forzado que viene dado por la secuencia transformada

j) generar el PRNG un dígito de flujo de clave de salida en función de la primera clave y el estado actual del PRNG

k) leer un dígito del texto simple

l) combinar el dígito de texto simple y el dígito de flujo de clave generando como resultado un dígito del texto cifrado, siendo la combinación una operación XOR del dígito de texto simple y el dígito de flujo de clave

m) almacenar el dígito de texto cifrado obtenido y añadirlo a la secuencia de texto cifrado previa, la secuencia de texto cifrado obtenida será la secuencia de texto cifrado actual

n) avanzar a la etapa d).

En un segundo aspecto, se presenta un sistema para mejorar la encriptación y desencriptación de datos por cifrado de flujo, comprendiendo el sistema un dispositivo de cifrado y un dispositivo de descifrado, en el que, en el dispositivo de cifrado, los datos que van a encriptarse, denominados texto simple, se reciben y se combinan con una secuencia aleatoria de dígitos, denominada flujo de clave, para obtener el texto simple encriptado denominado texto cifrado, y en el dispositivo de descifrado, el texto que va a desencriptarse, texto cifrado, se recibe y se combina con el flujo de clave para obtener el texto desencriptado, texto simple, siendo los dígitos bloques de s bits;

comprendiendo el dispositivo de cifrado:

- un generador de números pseudoaleatorios, PRNG, generándose el flujo de clave por el PRNG en función de una clave, K1, y el estado actual del generador y determinándose el siguiente estado del generador por K1 y el estado previo si no hay un valor forzado para el siguiente estado
- un combinador que lee un dígito del texto simple, lee un dígito del flujo de clave, combina ambos dígitos y genera como resultado un dígito del texto cifrado, siendo la combinación una operación XOR del dígito de texto simple y el dígito de flujo de clave
- una primera línea de retardo que retarda la secuencia de texto cifrado k dígitos, siendo k un parámetro de diseño, k≥0.
- una segunda línea de retardo que retarda la secuencia de texto cifrado j dígitos, siendo j un parámetro de diseño, j≥0.
- un primer registro de memoria que almacena los últimos l dígitos de las secuencias de texto cifrado retardadas por la primera línea de retardo, siendo l un parámetro de diseño, actualizándose el registro cada vez que se genera un nuevo dígito de texto cifrado
- un segundo registro de memoria que almacena los últimos m dígitos de las secuencias de texto cifrado retardadas por la segunda línea de retardo, siendo m un parámetro de diseño, actualizándose el registro cada vez que se genera un nuevo dígito de texto cifrado
- un módulo de transferencia que aplica una función de transferencia a la secuencia de m dígitos almacenados en el segundo registro de memoria, generando una secuencia transformada
- un comparador que, tras cada nuevo dígito generado del texto cifrado, compara la secuencia de l dígitos almacenados en el primer registro de memoria con una segunda clave K2 que es una secuencia de l dígitos; y si ambas secuencias coinciden, genera una orden de carga al PRNG para cambiar el estado del PRNG a un valor forzado que viene dado por la secuencia transformada generada por el módulo de transferencia;

comprendiendo el dispositivo de descifrado:

- un generador de números pseudoaleatorios, PRNG, generándose el flujo de clave por el PRNG en función de la clave K1 y el estado actual del generador y determinándose el siguiente estado del generador por K1 y el estado previo si no hay un valor forzado para el siguiente estado
- un combinador que lee un dígito del texto cifrado, lee un dígito del flujo de clave, combina ambos dígitos y genera como resultado un dígito del texto cifrado desencriptado, que es el texto simple, siendo la combinación una operación XOR del dígito de texto cifrado y el dígito de flujo de clave
- una primera línea de retardo que retarda la secuencia de texto cifrado k dígitos
- una segunda línea de retardo que retarda la secuencia de texto cifrado j dígitos
- un primer registro de memoria que almacena los últimos l dígitos de las secuencias de texto cifrado retardadas por la primera línea de retardo, actualizándose el registro cada vez que se recibe un nuevo

dígito de texto cifrado

- un segundo registro de memoria que almacena los últimos m dígitos de las secuencias de texto cifrado retardadas por la segunda línea de retardo, siendo m un parámetro de diseño, actualizándose el registro cada vez que se recibe un nuevo dígito de texto cifrado
- un módulo de transferencia que aplica una función de transferencia a la secuencia de m dígitos almacenados en el segundo registro de memoria, generando una secuencia transformada

un comparador que, tras cada nuevo dígito recibido del texto cifrado, compara la secuencia de l dígitos almacenados en el primer registro de memoria con la segunda clave K2; y si ambas secuencias coinciden, se genera una orden de carga al PRNG para cambiar el estado del PRNG a un valor forzado que viene dado por la secuencia transformada generada por el módulo de transferencia.

Finalmente, se presenta un programa informático que comprende medios de código de programa informático adaptados para realizar el procedimiento anteriormente descrito.

Para una comprensión más completa de la invención, sus objetos y ventajas, puede hacerse referencia a la siguiente memoria descriptiva y a los dibujos adjuntos.

BREVE DESCRIPCIÓN DE LOS DIBUJOS

Para completar la descripción y con el fin de proporcionar una mejor comprensión de la invención, se proporciona un juego de dibujos. Dichos dibujos constituyen una parte integrante de la descripción e ilustran una realización preferida de la invención, que no debe interpretarse como limitativa del alcance de la invención, sino más bien como un ejemplo de cómo puede realizarse la invención. Los dibujos comprenden las siguientes figuras:

La figura 1 representa un diagrama de bloques de un sistema de cifrado y descifrado de flujo síncrono.

La figura 2 representa un diagrama de bloques de un sistema de cifrado y descifrado de flujo de autosincronización.

La figura 3 representa un diagrama de bloques del extremo de cifrado del mecanismo de sincronismo aleatorio de cifrados de flujo.

La figura 4 representa un diagrama de bloques del extremo de descifrado del mecanismo de sincronismo aleatorio de cifrados de flujo.

La figura 5 representa un diagrama de bloques del mecanismo de sincronismo aleatorio de cifrados de flujo.

La figura 6 representa un diagrama de bloques de una realización de la invención que usa un registro de desplazamiento con realimentación lineal como generador de números pseudoaleatorios.

La figura 7 representa un diagrama de bloques de una realización de la invención que usa TDEA de cifrado por bloques en un modo de contador como generador de números pseudoaleatorios.

Los números de referencia y los símbolos correspondientes en las diferentes figuras se refieren a partes correspondientes a menos que se indique lo contrario.

DESCRIPCIÓN DETALLADA DE LA INVENCIÓN

Se propone un nuevo método y sistema para sincronizar cifrados de flujo, denominado sincronismo aleatorio de cifrados de flujo. A diferencia de los cifrados de flujo síncronos, que se sincronizan sólo una vez al comienzo de la transmisión, y de los cifrados de flujo de autosincronización, que se sincronizan siempre que se encripta un dígito; el nuevo método realiza una operación de sincronización a intervalos aleatorios, en una planificación temporal impredecible.

En la invención propuesta, la decisión de llevar a cabo la sincronización de las máquinas de cifrado y descifrado se toma con la aparición de una cadena predeterminada de dígitos de texto cifrado. Tal cadena de dígitos puede preseleccionarse de manera aleatoria, y formará parte de la clave de sistema. No obstante, su longitud debe seleccionarse en función de la frecuencia media deseada de sincronización. Además, el esquema puede aumentar la seguridad de la transmisión encriptada aumentando el tamaño de la clave de sistema.

Una condición necesaria para la seguridad de un procedimiento de encriptación es que la secuencia de texto cifrado sea completamente aleatoria. Por tanto, la aparición de cualquier cadena predeterminada en la secuencia de texto cifrado es completamente posible. La frecuencia media de la aparición de tal cadena depende de su longitud de bits, por motivos estadísticos, siendo su probabilidad de aparición inversamente proporcional a su longitud: aparecerán con frecuencia cadenas de pocos bits y aparecerán rara vez cadenas de muchos bits. Una cadena de *l* bits de longitud se produce cada 2^l bits, en promedio, en cualquier secuencia de bits aleatoria; por tanto, la probabilidad media de

aparición espontánea de cualquier cadena predeterminada de *l* bits de longitud, dentro de una secuencia aleatoria de bits, es igual a:

$$P_l = 1 / 2^l \qquad (1)$$

No todas las apariciones espontáneas de una cadena predeterminada se producirán con igual frecuencia dentro de una secuencia aleatoria de bits; la distancia *k,* en lugares de bit, entre sus apariciones puede variar desde un valor de distancia de $k = 0$, hasta distancias grandes, siendo las distancias crecientes progresivamente menos probables. La probabilidad de aparición de dos cadenas iguales de longitud *l* con una separación de *k* lugares de bit es de:

$$P_{l,k} = (1 / 2^l) \cdot (1 / 2^l) \cdot (1 - 1 / 2^l)^k. \qquad (2)$$

Distancias muy grandes entre cadenas iguales se producen rara vez. La probabilidad conjunta total de apariciones de cadenas iguales de longitud *l*, con distancias $k > d$ es de:

$$\sum P_{l,d} = (1 / 2^l) \cdot (1 - 1 / 2^l)^d = P_l \cdot (1 - 1 / 2^l)^d. \qquad (3)$$

Como ejemplo, supóngase que $l = 8$, por tanto $2^l = 256$, la probabilidad de apariciones de cadenas con una separación $k > 256$ es de: $\sum P_{8,256} < 0{,}0014$.

El sincronismo se ejecuta cada vez que se observa la aparición espontánea de una cadena de una longitud de *l* dígitos predeterminados; cuando se produce esta sucesión de dígitos de texto cifrado se realiza la sincronización, cambiando el estado del PRGN que generó el flujo de clave, siendo el nuevo estado una función no lineal de un conjunto de dígitos transmitidos previamente.

La figura 3 representa el diagrama de bloques del mecanismo de sincronismo aleatorio de cifrados de flujo, en la ubicación del emisor; es decir, el extremo de cifrado; nuevos bloques específicos de esta invención, no usados en cifrados de flujo síncronos ni en cifrados de flujo de autosincronización, se distinguen con un borde discontinuo. Como en cualquier cifrado de flujo convencional, hay un texto simple (3a) que debe encriptarse para dar el texto cifrado (5a), mediante la combinación (4a) de los bits de texto simple con los bits de flujo de clave generados por el PRNG (2a); este PRNG genera un flujo de clave que es una función complicada de la clave de sistema *K*1, almacenada en el registro (1a) y del estado del PRNG. La combinación (4a) consiste en XOR, si tanto el texto simple como el flujo de clave se componen de bits independientes, la operación XOR se lleva a cabo bit a bit; pero si tanto el texto simple como el flujo de clave se componen de dígitos formados por bloques de varios bits, la operación XOR se realiza a nivel de bits. El mecanismo de sincronización se logra mediante el resto de los elementos de la figura 3, de la siguiente manera:

El texto cifrado (5a) además de enviarse al extremo receptor, también se encamina en paralelo a las dos líneas de retardo (13a) y (14a), cuyas longitudes de retardo *k* y *j* pueden estar comprendidas entre 0 y cualquier valor. El texto cifrado, tras pasar a través de las líneas de retardo (13a) y (14a), llega a dos registros de memoria (15a) y (16a), con una capacidad de *l* dígitos y *m* dígitos, respectivamente. Los valores de longitud de retardo *k* y *j* deben seleccionarse de manera que los dígitos almacenados en el registro (15a) y el registro (16a) no coincidan.

El contenido de los registros (15a) y (16a) se actualiza cada vez que se genera un nuevo dígito de texto cifrado. Este dígito de texto cifrado entra en las líneas de retardo (13a) y (14a) sustituyendo al dígito introducido previo. El último dígito de cada línea de retardo (13a) y (14a) sale y entra en los registros (15a) y (16a) respectivamente.

Tras cada nuevo dígito generado de texto cifrado, el contenido *L* del registro (15a) se compara con la clave *K*2, almacenada en el registro (17a), la comparación se realiza mediante el comparador (18a).

Si el $L \neq K2$, es decir el contenido de ambos registros (17a) y (15a) no coincide, no se emprende ninguna acción; por tanto, el PRNG (2a) continúa inalterado generando nuevos dígitos de clave. Pero si el contenido de los registros (17a) y (15a) sí concuerda, $L = K2$, el comparador (18a) envía una orden de carga, al PRNG (2a), de saltar de manera forzada a un nuevo estado. El valor del nuevo estado del PRNG (2a) se carga desde la salida del estado de función de transferencia (11a), que es una transformación $q(M)$ de la cadena de texto cifrado *M,* contenido almacenado en el registro (16a). Tal transformación puede ser cualquier función, que puede estar definida por completo o puede depender de una tercera clave *K*3 (19a); y una función de este tipo también puede ser una identidad.

La figura 4 representa el diagrama de bloques del mecanismo de sincronismo aleatorio de cifrados de flujo, en la ubicación del receptor, es decir en el extremo de descifrado; nuevos bloques específicos de esta invención, no usados en cifrados de flujo síncronos ni en cifrados de flujo de autosincronización, se distinguen con un borde no discontinuo. El diagrama de bloques es similar al diagrama del extremo de cifrado, pero con tres diferencias. En primer lugar, el bloque (3b) consiste en texto cifrado entrante. En segundo lugar, el bloque (5b) consiste en texto simple recuperado, que es el resultado de la combinación (4b) de los dígitos de texto cifrado con los dígitos de flujo de clave generados por el PRNG (2b). En tercer lugar, la conexión de entrada de ambas líneas de retardo (13b) y (14b) se realiza antes de la combinación (4b), por medio de XOR, de los dígitos del texto cifrado (3b) y los dígitos de flujo de clave generados por el PRNG (2b). El resto de los elementos del diagrama de bloques son iguales que los del extremo de cifrado, es decir, las líneas de retardo 13b y 14b que retardan el texto cifrado k y j dígitos, respectivamente; los registros

15b y 16b para almacenar las señales de retardo; un comparador 18b que compara la primera señal de retardo con K2 almacenada en el registro 17b y envía una orden de carga de saltar a un nuevo estado al PRNG si las dos señales concuerdan. Dicho nuevo estado se carga desde la salida del estado de función de transferencia 11b y puede depender de una tercera clave K3 almacenada en un registro (13b).

La figura 5 ilustra el diagrama de flujo del mecanismo de sincronismo aleatorio de cifrados de flujo; las etapas 20, 21, 22, 23, 24, 25, 26, 27, 28 y 29, son las etapas normales de cualquier cifrado de flujo síncrono. El resto son las nuevas etapas que se añaden para convertirlo en un cifrado de flujo con sincronismo aleatorio.

La etapa 20 consiste en inicializar el PRNG, determinando un estado inicial con el valor inicial *i*. La etapa 21 consiste en leer del registro 22 la clave *K*1. La etapa 23 consiste en avanzar sin alteraciones al siguiente estado del PRNG cada vez que va a generarse un dígito de flujo de clave según la función de transición de estados *f* (K1, *S*) que lo controla. La etapa 24 consiste en generar un dígito de flujo de clave de salida aplicando la función g a la clave *K*1 y al estado generado. La etapa 25 consiste en leer un dígito de texto simple del almacén 26. La etapa 27 consiste en combinar por medio de la operación XOR el dígito de texto simple leído en la etapa 25 con el dígito de flujo de clave generado en la etapa 24. En la etapa 28 se obtiene el texto cifrado y se almacena en el registro 29.

Las etapas pertenecientes al mecanismo de sincronismo aleatorio de cifrados de flujo se describen a continuación y se muestran en la figura 5 con un borde discontinuo.

Con el fin de sincronizar inicialmente los mecanismos de cifrado y descifrado, la etapa 20 además de inicializar el PRNG en un estado inicial con el valor inicial *i*, reajusta al estado cero el contenido de las líneas de retardo (13a y 14a) y los correspondientes registros (15a y 16a) del extremo de cifrado así como las líneas de retardo (13b, 14b) y los registros (15b y 16b) del descifrado.

La etapa 30 consiste en leer la clave *K*2, de *l* dígitos de tamaño, del almacén 31. La etapa 32 consiste en leer del almacén 29 una cadena de *l* dígitos de texto cifrado *L*, (tras un retardo de *k* dígitos). La etapa 33 consiste en comparar la clave *K*2 con el texto cifrado retardado *L*; en este punto se toma una decisión: si ambas cantidades son diferentes, el sistema continúa a la etapa 23 y siguientes; pero si ambas cantidades concuerdan, el sistema continúa a la etapa 34.

La etapa 34 consiste en leer una cadena de *m* dígitos de texto cifrado *M*, del almacén 29, tras un retardo de *j* dígitos. La etapa 35 consiste en transformar los m dígitos de texto cifrado M para obtener el siguiente estado futuro *q*(*M*) del PRNG. La etapa 36 consiste en forzar el siguiente estado del PRNG para que adopte el valor de la función transformada *q*(*M*). Entonces el sistema continúa a la etapa 24 y siguientes.

Existen muchas realizaciones posibles de la invención, dependiendo del PRNG usado y la función de transferencia del mecanismo de sincronismo aleatorio del cifrado de flujo. Como ejemplo se presentan dos realizaciones distintas.

Cifrado de flujo usando un registro de desplazamiento con realimentación lineal como PRNG

El diagrama de bloques de la realización propuesta se ilustra en la figura 6, que es similar al diagrama de bloques representado en la figura 3. La numeración de los bloques es la misma en ambas figuras, pero en lugar del sufijo “a” utilizado en la figura 3, en la figura 6 se usa el sufijo “c”.

El PRNG (2c) consiste ahora en un registro de desplazamiento con realimentación lineal (LFSR), con un longitud de 127, y polinomio característico $x^{127} + x^{63} + 1$ (el mayor exponente se denomina *grado del polinomio*). La clave *K*1 consiste en el valor de estado inicial del registro de desplazamiento. El valor de los exponentes del polinomio característico puede interpretarse como una clave estructural, que no se cambia frecuentemente; debe seleccionarse para garantizar el máximo periodo de repetición admisible, lo que se obtiene seleccionando un polinomio irreducible primitivo. En la presente realización, para obtener la máxima eficacia computacional, el polinomio seleccionado es simplemente un trinomio, precisamente uno primitivo cuyo grado *p* es un número primo de Mersenne (todos los polinomios de grado primo de Mersenne son irreducibles).

La longitud seleccionada de la línea de retardo equivalente a (13a) es $k = 0$, es decir, no existe. La longitud del registro (15c), del comparador (18c) y de la clave *K*2 es $l = 6$ bits. La longitud de la línea de retardo (14c) es $j = 6$ bits, de esta manera se garantiza que el contenido *L* y *M* de los registros (15c) y (16c) corresponden a partes diferentes del texto cifrado. La longitud del registro (16c) es de 128 bits. Cada vez que coinciden el contenido *L* del registro (15c) y la clave *K*2, el comparador (18c) genera una nueva orden de carga de estado para el PRNG.

La función de transferencia (11c) consiste en cifrado por bloques de AES, cuya longitud de bloque es de 128 bits, y su clave *K*3, almacenada en el registro (19c), se selecciona con una longitud de 256 bits. La salida de la función de transferencia $AES_{K3}(M)$ se carga en el LFSR del PRNG, como un nuevo valor forzado de estado, cada vez que el comparador (18c) genera una nueva orden de carga de estado. Como la salida del AES es un bloque de 128 bits y el LFSR tiene 127 bits de longitud, el bit más significativo del bloque de salida del AES se desecha, antes de cargarlo en el LFSR.

Un PRNG realizado con un LFSR tiene propiedades de aleatoriedad perfecta pero desafortunadamente no es seguro, como debiera ser. La forma de quebrantar un cifrado de flujo, que usa un LFSR como PRNG, consiste en determinar el valor del polinomio característico del LFSR y su presente estado. Para determinarlos, basta con conocer un fragmento del flujo de clave generado, de longitud doble a la longitud del registro de desplazamiento. Un enemigo puede llegar a conocer un fragmento de este tipo del flujo de clave por medio de un "ataque de texto simple conocido". No obstante, cuando se añade el mecanismo de sincronismo aleatorio, objeto de esta invención, esta clase de ataque es infructuosa, porque un enemigo no puede tener acceso a la cadena inalterada de 254 bits, necesaria para montar el ataque. Debido a la aparición aleatoria de las coincidencias entre la clave *K*2 y el contenido *L* del registro (15c), la distancia media entre apariciones es de 2^l = 64 bits; pero según la ec. (3) la proporción de apariciones con una separación mayor de 254 bits es de sólo $1 \cdot 10^{-3}$ de las apariciones totales. Por tanto, un oponente tendrá una probabilidad muy baja de hallar una cadena inalterada de 254 bits, necesaria para determinar el polinomio característico y el estado del LFSR. Además, si tuviera éxito al hallar el polinomio característico y el estado del LFSR, no podría sacar provecho de ello, porque el estado del LFSR cambiará muy pronto a un nuevo valor forzado inesperado.

El LFSR generará, si se mantiene inalterado, un flujo de clave ideal de 2^{127}–1 bits de longitud. El efecto neto de la invención es equivalente a extraer diferentes segmentos, al azar, de longitud arbitraria, del flujo de clave ideal mencionado anteriormente, para construir otro flujo de clave, encadenándolos entre sí dispuestos en un orden aleatorio completamente nuevo. La realización propuesta tiene una fuerza mucho mayor que el cifrado de flujo original sin sincronismo aleatorio, porque además de conocer el estado inicial del registro de desplazamiento y el polinomio característico, el atacante debe conocer las claves *K*2 y *K*3 para quebrantarlo.

Cifrado de flujo usando TDEA de cifrado por bloques en modo de contador como PRNG

El diagrama de bloques de la realización propuesta se ilustra en la figura 7, que es similar al diagrama de bloques representado en la figura 3. La numeración de los bloques es la misma en ambas figuras, pero en lugar del sufijo "a" utilizado en la figura 3, en la figura 6 se usa el sufijo "d".

El PRNG (2d) consiste ahora en un TDEA de cifrado por bloques en modo de contador, siendo su longitud de bloque de 64. La clave (1d) se compone del estado inicial del contador *i* y de la *K*1 del cifrado por bloques de TDEA. El contador es un contador binario de 64 bits simple. La longitud seleccionada de la línea de retardo equivalente a (13a) es *k* = 0, es decir, no existe. La longitud del registro (15c), del comparador (18c) y de la clave *K*2 es *l* = 8 bits. La longitud de la línea de retardo (14c) es *j* = 8 bits y la longitud del registro m=64 bits, de esta manera se garantiza que el contenido *L* y *M* de los registros (15c) y (16c) corresponde a partes diferentes del texto cifrado.

Cada vez que coinciden el contenido *L* del registro (15c) y la clave *K*2, el comparador (18c) genera una nueva orden de carga de estado para el contador del PRNG, lo que provoca que el contador cambie su recuento, saltando a un nuevo recuento de valor completamente aleatorio, mayor o menor que el original. La función de transferencia equivalente a (11a) es simplemente una identidad, es decir, no existe. El contenido del registro (16d) se carga en el contador del PRNG, como un nuevo valor forzado de estado, cada vez que el comparador (18c) genera una nueva orden de carga de estado. Como consecuencia de la no existencia de la función de transferencia no hay tampoco ninguna clave *K*3.

Un PRNG realizado con un TDEA en el modo de contador genera una permutación aleatoria, bastante diferente de una secuencia pseudoaleatoria, porque las secuencias pseudoaleatorias muestran una gran cantidad de números repetidos (colisiones), mientras que las permutaciones aleatorias no tienen colisiones, es decir, todos los números generados son diferentes. Para evitar un ataque de distinción contra un PRNG realizado con un TDEA en el modo de contador, la longitud de la secuencia generada debe limitarse a un tamaño mucho menor que la "cota del cumpleaños", que en el caso de un cifrado por bloques de 64 bits es de sólo 2^{32} bits. No obstante, cuando se añade el mecanismo de sincronismo aleatorio, objeto de esta invención, un PRNG realizado con un TDEA en modo de contador sí que genera secuencias pseudoaleatorias auténticas. El motivo es que el efecto neto de la invención es equivalente a extraer diferentes segmentos, al azar, de longitud arbitraria, de la permutación aleatoria mencionada anteriormente, para construir otro flujo de clave diferente, encadenándolos entre sí dispuestos en un orden aleatorio completamente nuevo; este nuevo flujo de clave tendrá ahora colisiones, debido a la posibilidad de haber seleccionado dos veces (o más) algunas partes de la permutación aleatoria original. Por tanto, la longitud de la secuencia pseudoaleatoria utilizable puede ser tan grande como se desee, porque ya no está el límite de la cota del cumpleaños. La longitud de la secuencia generada puede ser incluso mayor que el periodo de repetición del contador inalterado, es decir, en la presente realización puede ser mayor que 2^{64} bits.

La realización propuesta tiene una fuerza mucho mayor que el cifrado de flujo original sin sincronismo aleatorio, porque además de conocer el estado inicial *i* del contador y la clave *K*1, el atacante debe conocer la clave *K*2 para quebrantarlo.

En resumen, las ventajas de la presente invención son:

- El mecanismo de sincronismo aleatorio de los cifrados de flujo permite que el sincronismo de un cifrado de flujo tenga lugar en puntos de tiempo aleatorios, por lo que un oponente no puede deducir el momento en el que tiene lugar el sincronismo, mejorando la seguridad del sistema.

- Añadiendo una nueva clave *K*2 se mejora la seguridad del sistema, debido a que un enemigo, cuando realiza un ataque de fuerza bruta (probando todos los valores de clave posibles), necesitará probar 2^{l} veces más valores de clave de lo que era necesario sin el mecanismo de sincronismo aleatorio.

- La función de transferencia *q*(*M*) puede incorporar una tercera clave *K*3 en su diseño, permitiendo un aumento complementario de la seguridad, debido al aumento de la dificultad de un ataque de fuerza bruta. En tal caso, un enemigo necesitará probar $2^{(l+m)}$ veces más valores de clave de lo que era necesario sin el mecanismo de sincronismo aleatorio, por lo que tardaría mucho más tiempo en "romper" el código.

- La función de transferencia *q*(*M*) puede convertir un cifrado de flujo inseguro, que hace uso de un PRNG predecible (como un registro de desplazamiento lineal con realimentación), en un cifrado de flujo seguro porque el flujo de clave resultante ya no es predecible debido a los saltos forzados aleatorios del generador.

- Los PRNG de algunos cifrados de flujo no generan un flujo de clave pseudoaleatorio auténtico, sino simplemente una permutación dependiente de la clave de todos los posibles estados del generador. Éste es el caso de un cifrado de flujo que funciona en el *modo de contador* o un generador congruencial lineal. Es posible recuperar el flujo de clave de tales cifrados de flujo por medio de un ataque de texto simple conocido, y montar entonces un ataque de distinción contra el PRNG, observando que la secuencia de flujo de clave generada no muestra colisiones (repetición de valores), como debe mostrar una verdadera secuencia pseudoaleatoria. La combinación de estos dos ataques puede permitir que un enemigo adivine, con cierta probabilidad de éxito, valores futuros del flujo de clave.

- El uso del mecanismo de sincronismo aleatorio convierte el flujo de clave de cualquier cifrado de flujo, construido con un PRGN que genera una permutación dependiente de la clave de todos los posibles estados, en un verdadero flujo de clave pseudoaleatorio, con colisiones; impidiendo, por tanto, el ataque descrito previamente.

- Los ataques de reproducción son perceptibles y se detectan más fácil y rápidamente. Con el uso del mecanismo de sincronismo aleatorio, cuando un enemigo graba algunos bits de texto cifrado y, en un momento posterior, sustituye esta grabación en el tráfico actual, induce una información inservible perceptible mucho más larga en el texto simple recuperado (mientras se resincroniza el extremo receptor) que en el caso de usar un cifrado de flujo de autosincronización, permitiendo la detección del ataque. Esto se debe a que el sincronismo aleatorio no tiene lugar cada dígito de texto cifrado (como en el sistema de autosincronización) sino cada 2^{l} dígitos de texto cifrado.

Aunque la presente invención se ha descrito en referencia a realizaciones específicas, los expertos en la técnica deben entender que pueden realizarse los anteriores y diversos otros cambios, omisiones y adiciones en la forma y detalle de las mismas, sin apartarse del sentido y alcance de la invención tal como se define en las reivindicaciones adjuntas.

REIVINDICACIONES

1. Un método para mejorar la encriptación de datos por cifrado de flujo, en el que los datos que van a encriptarse, denominados texto simple, se combinan con una secuencia aleatoria de dígitos, denominada flujo de clave, para obtener el texto simple encriptado denominado texto cifrado, generándose el flujo de clave mediante un generador de números pseudoaleatorios, PRNG, en función de una determinada secuencia de clave y el estado actual del generador, siendo los dígitos bloques de s bits, siendo s un parámetro de diseño, comprendiendo el método las siguientes etapas:

a) determinar el estado inicial del generador de números pseudoaleatorios y poner al estado cero, el contenido de una primera y una segunda línea de retardo (13a, 14a) del texto cifrado y el contenido de un primer y un segundo registro

b) obtener la primera clave, K1, que se usará como la secuencia de clave del PRNG

c) obtener una segunda clave, K2, que será una secuencia de l dígitos, siendo l un parámetro de diseño

d) retardar en la primera línea de retardo la secuencia actual del texto cifrado k dígitos y almacenar en el primer registro los últimos l dígitos del texto cifrado retardado, siendo k un parámetro de diseño, siendo k≥0.

e) comparar K2 con la secuencia de l bits del texto cifrado retardado almacenado en la etapa d)

f) si ambas secuencias no coinciden, generar el siguiente estado del PRNG en función de la primera clave y el estado previo y avanzar a la etapa j)

d) si ambas secuencias coinciden, retardar en la segunda línea de retardo el texto cifrado j dígitos y almacenar en el segundo registro los últimos m dígitos del texto cifrado retardado, siendo j y m parámetros de diseño, siendo j≥0

h) aplicar una función de transferencia a la secuencia de m dígitos almacenados en la etapa g) para obtener una secuencia transformada

i) generar una orden de carga al PRNG para cambiar el estado del PRNG a un valor forzado que viene dado por la secuencia transformada

j) generar el PRNG un dígito de flujo de clave de salida en función de la primera clave y el estado actual del PRNG

k) leer un dígito del texto simple

l) combinar el dígito de texto simple y el dígito de flujo de clave generando como resultado un dígito del texto cifrado, siendo la combinación una operación XOR del dígito de texto simple y el dígito de flujo de clave

m) almacenar el dígito de texto cifrado obtenido y añadirlo a la secuencia de texto cifrado previa, la secuencia de texto cifrado obtenida será la secuencia de texto cifrado actual

n) avanzar a la etapa d).

2. Un método según cualquiera de las reivindicaciones anteriores, en el que el estado inicial del PRNG viene dado por un valor inicial, i, que es un parámetro de diseño.

3. Un método según cualquiera de las reivindicaciones anteriores, en el que s=1, es decir, los dígitos son bits independientes.

4. Un método según cualquiera de las reivindicaciones 1-2, en el que, si s>1, la operación XOR se realiza en cada dígito bit a bit.

5. Un método según cualquiera de las reivindicaciones anteriores, en el que las etapas de obtener las claves primera y segunda comprenden las etapas de leer los registros correspondientes en los que están almacenadas las claves.

6. Un método según cualquiera de las reivindicaciones anteriores, en el que la función de transferencia depende de una tercera clave, K3, almacenada en un registro.

7. Un método según las reivindicaciones 1 a 5, en el que la función de transferencia es una identidad.

8. Un método según cualquiera de las reivindicaciones anteriores, en el que $k > j + m$.

9. Un método según las reivindicaciones 1 a 7, en el que $j > k + l$.

10. Un sistema para mejorar la encriptación y desencriptación de datos por cifrado de flujo, comprendiendo el sistema un dispositivo de cifrado y un dispositivo de descifrado, en el que, en el dispositivo de cifrado, los datos que van a encriptarse, denominados texto simple, se reciben y se combinan con una secuencia aleatoria de dígitos, denominada flujo de clave, para obtener el texto simple encriptado denominado texto cifrado, y en el dispositivo de descifrado, el

texto que va a desencriptarse, texto cifrado, se recibe y se combina con el flujo de clave para obtener el texto desencriptado, texto simple, siendo los dígitos bloques de s bits;

comprendiendo el dispositivo de cifrado:

- un generador de números pseudoaleatorios, PRNG, generándose el flujo de clave por el PRNG en función de una clave, K1, y el estado actual del generador y determinándose el siguiente estado del generador por K1 y el estado previo si no hay un valor forzado para el siguiente estado

- un combinador que lee un dígito del texto simple, lee un dígito del flujo de clave, combina ambos dígitos y genera como resultado un dígito del texto cifrado, siendo la combinación una operación XOR del dígito de texto simple y el dígito de flujo de clave

- una primera línea de retardo que retarda la secuencia de texto cifrado k dígitos, siendo k un parámetro de diseño, k≥0.

- una segunda línea de retardo que retarda la secuencia de texto cifrado j dígitos, siendo j un parámetro de diseño, j≥0.

- un primer registro de memoria que almacena los últimos l dígitos de las secuencias de texto cifrado retardadas por la primera línea de retardo, siendo l un parámetro de diseño, actualizándose el registro cada vez que se genera un nuevo dígito de texto cifrado

- un segundo registro de memoria que almacena los últimos m dígitos de las secuencias de texto cifrado retardadas por la segunda línea de retardo, siendo m un parámetro de diseño, actualizándose el registro cada vez que se genera un nuevo dígito de texto cifrado

- un módulo de transferencia que aplica una función de transferencia a la secuencia de m dígitos almacenados en el segundo registro de memoria, generando una secuencia transformada

- un comparador que, tras cada nuevo dígito generado del texto cifrado, compara la secuencia de l dígitos almacenados en el primer registro de memoria con una segunda clave K2 que es una secuencia de l dígitos; y si ambas secuencias coinciden, genera una orden de carga al PRNG para cambiar el estado del PRNG a un valor forzado que viene dado por la secuencia transformada generada por el módulo de transferencia;

comprendiendo el dispositivo de descifrado:

- un generador de números pseudoaleatorios, PRNG, generándose el flujo de clave por el PRNG en función de la clave K1 y el estado actual del generador y determinándose el siguiente estado del generador por K1 y el estado previo si no hay un valor forzado para el siguiente estado

- un combinador que lee un dígito del texto cifrado, lee un dígito del flujo de clave, combina ambos dígitos y genera como resultado un dígito del texto cifrado desencriptado, que es el texto simple, siendo la combinación una operación XOR del dígito de texto cifrado y el dígito de flujo de clave

- una primera línea de retardo que retarda la secuencia de texto cifrado k dígitos

- una segunda línea de retardo que retarda la secuencia de texto cifrado j dígitos

- un primer registro de memoria que almacena los últimos l dígitos de las secuencias de texto cifrado retardadas por la primera línea de retardo, actualizándose el registro cada vez que se recibe un nuevo dígito de texto cifrado

- un segundo registro de memoria que almacena los últimos m dígitos de las secuencias de texto cifrado retardadas por la segunda línea de retardo, siendo m un parámetro de diseño, actualizándose el registro cada vez que se recibe un nuevo dígito de texto cifrado

- un módulo de transferencia que aplica una función de transferencia a la secuencia de m dígitos almacenados en el segundo registro de memoria, generando una secuencia transformada

- un comparador que, tras cada nuevo dígito recibido del texto cifrado, compara la secuencia de l dígitos almacenados en el primer registro de memoria con la segunda clave K2; y si ambas secuencias coinciden, se genera una orden de carga al PRNG para cambiar el estado del PRNG a un valor forzado que viene dado por la secuencia transformada generada por el módulo de transferencia.

11. Un programa informático que comprende medios de código de programa informático adaptados para realizar el procedimiento según cualquiera de las reivindicaciones 1 a 9, cuando dicho programa se ejecuta en un ordenador, un procesador de señal digital, una disposición de puertas programables en campo, un circuito integrado de aplicación específica, un microprocesador, un microcontrolador o cualquier otra forma de hardware programable.

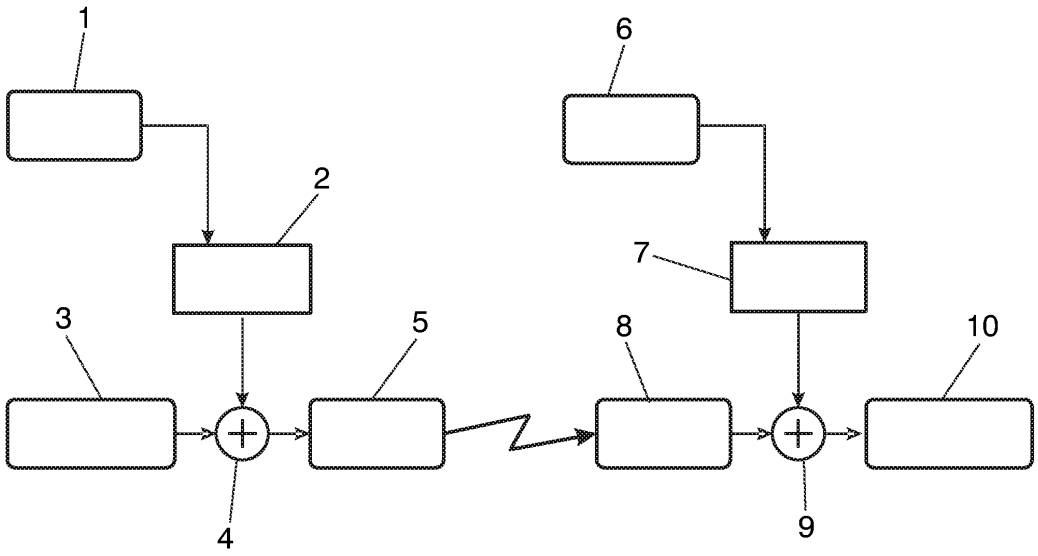

FIG. 1

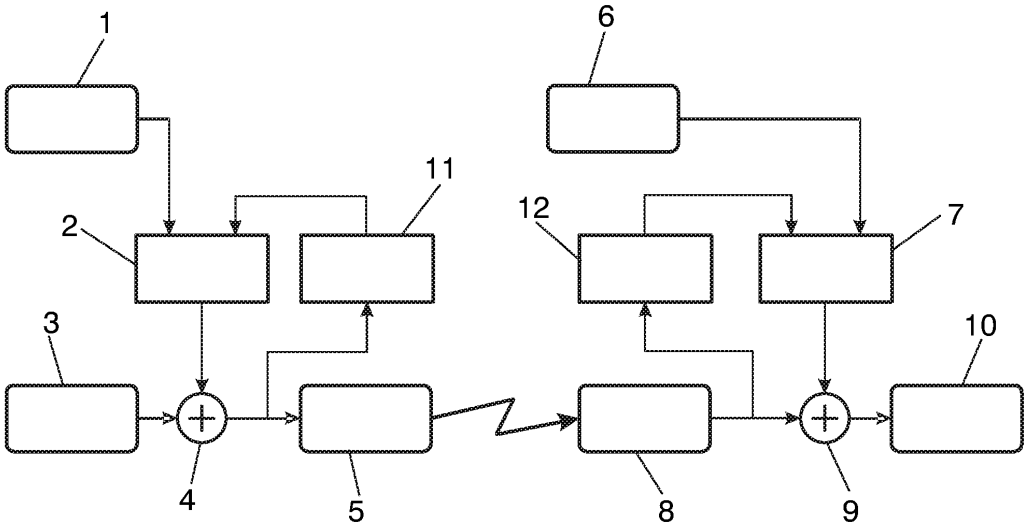

FIG. 2

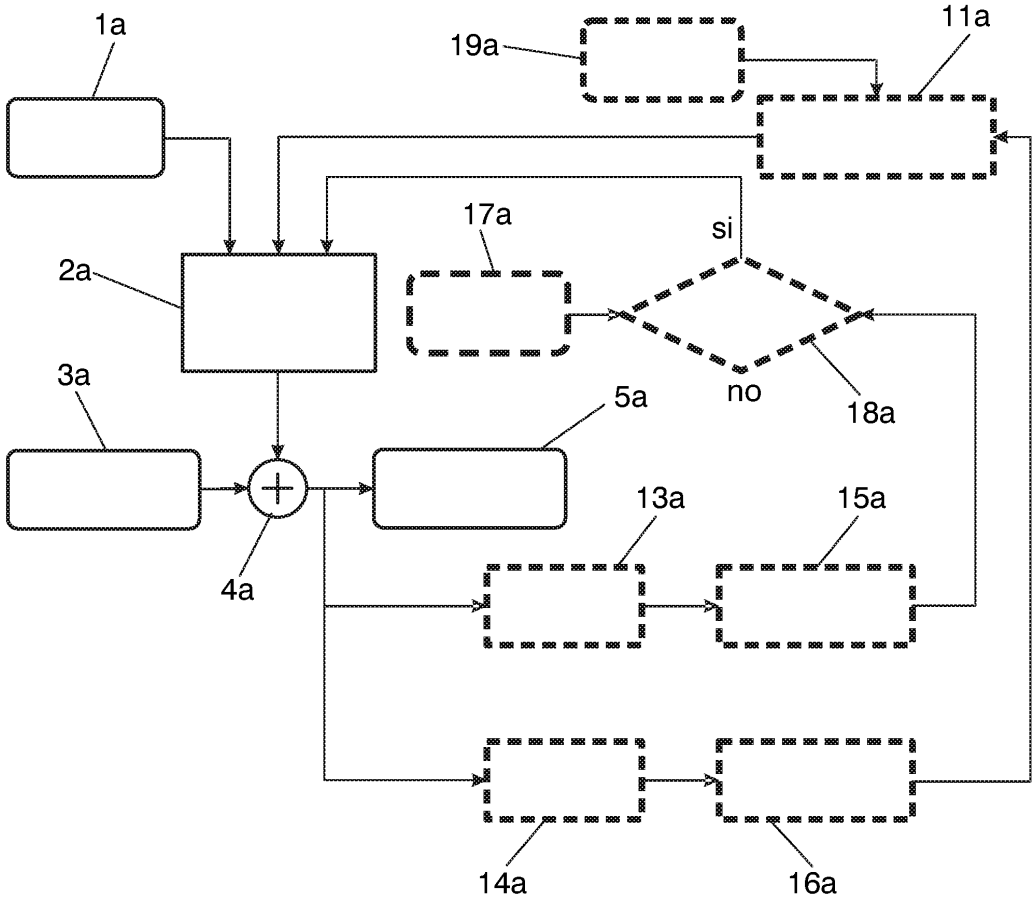

FIG. 3

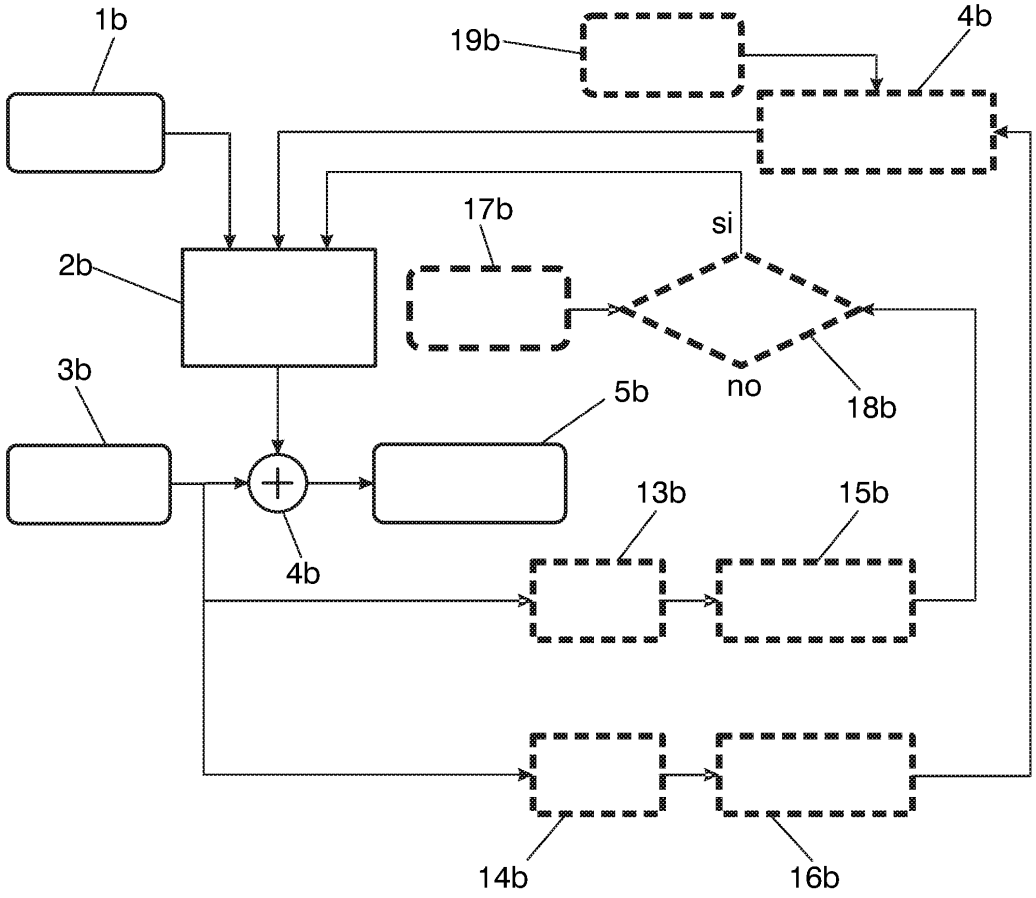

FIG. 4

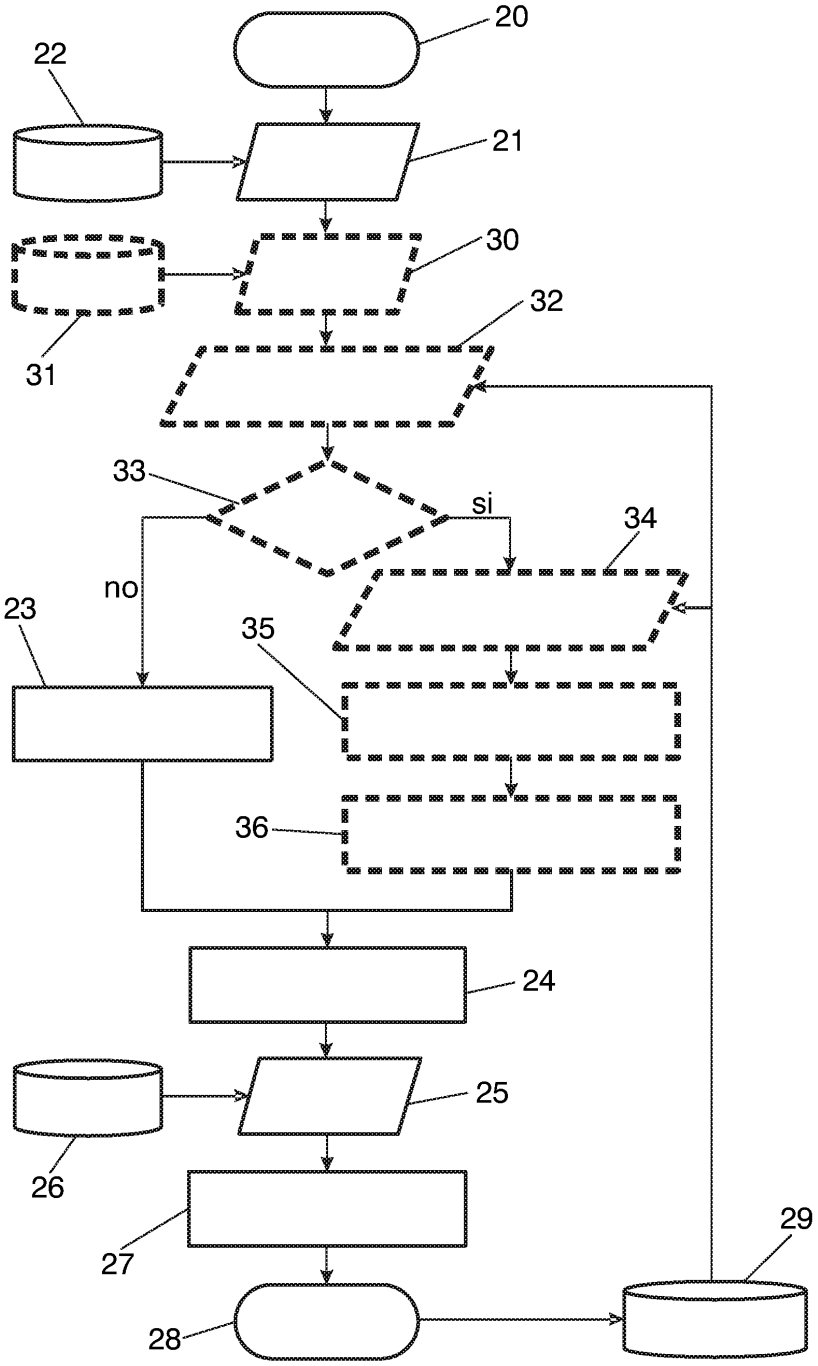

FIG. 5

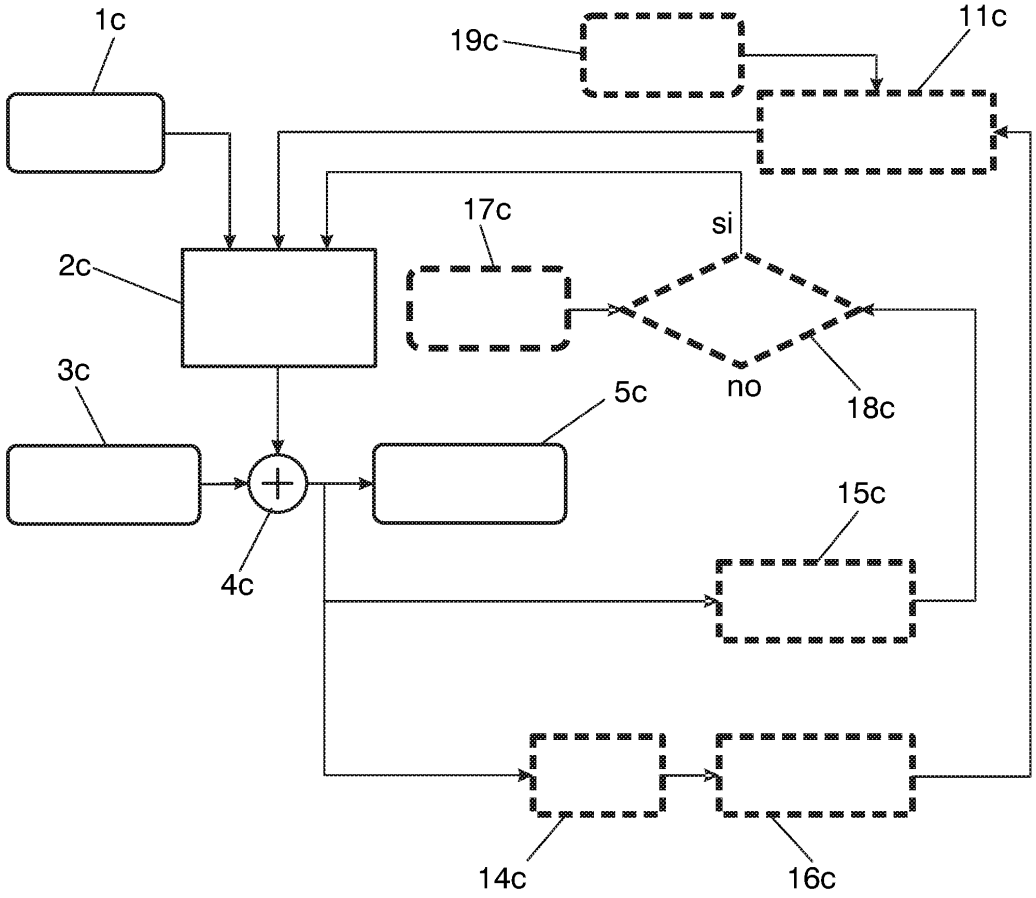

FIG. 6

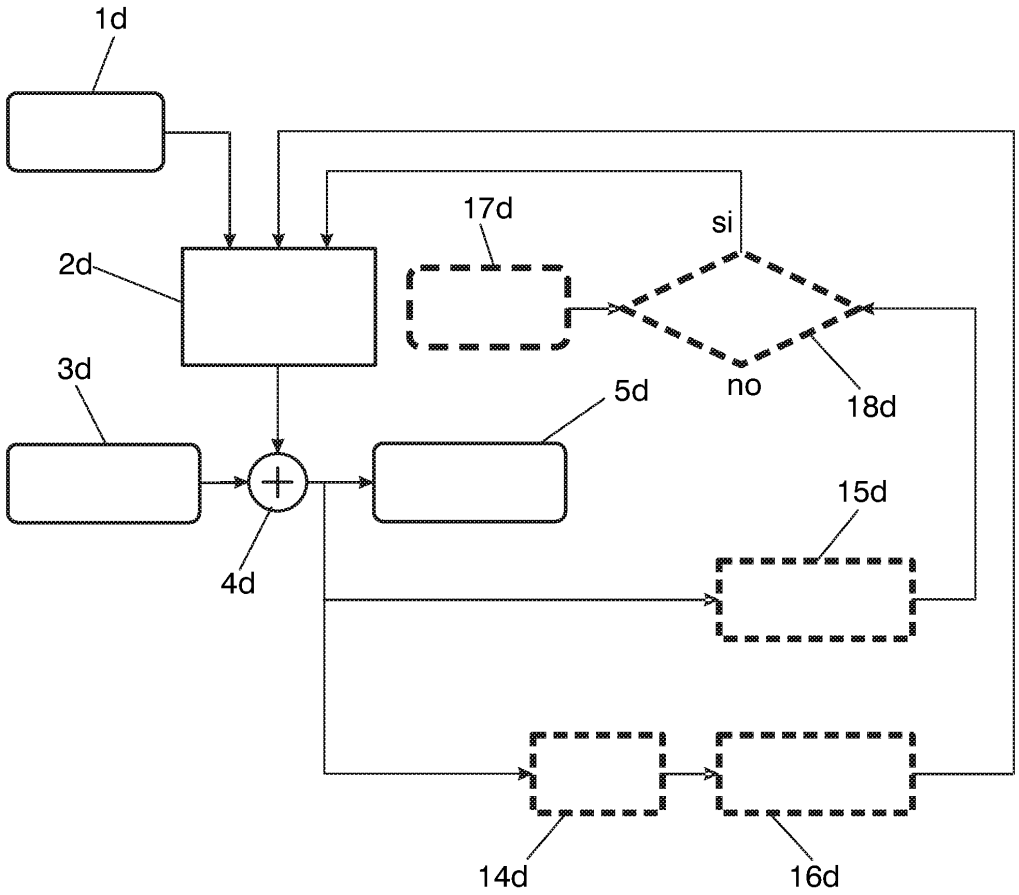

FIG. 7