# STATE OF THE ART

En un mundo cada vez más dominado por un IoT que, si bien podríamos considerar en fase relativamente temprana, ya ha avanzado hasta cuotas que hace unos años eran inimaginables, nos encontramos con ciertas convenciones sobre criptografía ligera aplicada a estos dispositivos de usos cotidianos.

Estos dispositivos tienen la necesidad de funcionar con procesadores reducidos y de baja potencia con el fin de primar su utilidad cotidiana por encima de su complejidad, lo que deja poco espacio para desarrollar medidas de seguridad en la mayoría de los casos. Cabe destacar el caso de los dispositivos basados en etiquetas RFID como podrían ser las pulseras de acceso a centros deportivos; estos dispositivos contienen información que debe protegerse, a pesar de ser tan sencillos en su construcción que no poseen ni batería propia, sino que se alimentan de la propia energía del lector cuando las acercamos.

Hay que tener en cuenta que los sistemas criptográficos modernos, en muchos casos requieren de una potencia computacional demasiado elevada para poder implementarse en estos dispositivos, que, por otra parte, necesitan poder encriptar información sensible y personal con el fin de evitar filtraciones de datos, suplantación de usuarios…

Esta tarea es compleja cuando tratamos de aplicarla a dispositivos basados en microprocesadores que en muchos casos no superan los 16 bytes de memoria RAM para un procesador de 4, 8 o 16 bits.

De esta forma, un algoritmo AES o RSA sería prácticamente imposible de implementar en uno de estos dispositivos, lo que nos sitúa constantemente en la tesitura de tener que encontrar un equilibrio entre ligereza y seguridad. A más seguro sea el algoritmo, peor será el funcionamiento del dispositivo o más potencia necesitará, lo que repercute directamente de forma negativa en el producto, que, por otro lado, mientras menos potente sea su encriptación, más peligroso vuelve su uso y peor imagen dará de él.

Esta necesidad de seguridad, que a primera vista puede pensarse comúnmente que solo afecta a dispositivos como relojes digitales, hay que sumarle que cada día más dispositivos como bombas de insulina, marcapasos, wearables de todo tipo, sistemas de peaje, lectores de tarjetas de crédito contactless…

Para empezar a profundizar podemos enumerar los algoritmos de cifrado ligero más utilizados actualmente para dispositivos enfocados al IoT, que serán analizados en el siguiente capítulo.

* Cifrado en bloque
  + PRESENT
  + SPECK
  + SIMON
* Funciones Hash
  + PHOTON
  + QUARK
* Cifrado en flujo
* Grain
* Trivium
* Mickey
* MAC (Código de autenticación de mensajes)
* Chaskey
* TuLP
* LightMAC

# Cifrado en bloque

Estos cifrados ligeros se han diseñado con el objetivo de poder crear un sustituto de AES que pueda funcionar de forma eficiente en dispositivos con poca capacidad de procesamiento, pero siempre con el objetivo de poder mantener un nivel de seguridad lo más cercano posible a pesar de trabajar con bloques más pequeños.

En esta sección trataremos algunos algoritmos ligeros de cifrado en bloque que he considerado interesantes a la hora de una posible implementación en dispositivos inteligentes de baja potencia.

## Present

El algoritmo de cifrado Present fue desarrollado en 2007 por Orange Labs junto a la Universidad de Bochum y la Universidad Técnica de Dinamarca y está enfocado a etiquetas RFID.

La seguridad de estas etiquetas en el IoT es muy importante, ya que son estas las que permiten la identificación inequívoca de un equipo, y el no preocuparse por protegerlas puede dar lugar a ataques de suplantación.

Present es casi tres veces más ligero que AES y está pensado para aparatos con un consumo de energía muy bajo que necesiten una gran eficiencia. Ha sido incluido en el nuevo standard internacional de métodos criptográficos ligeros.

Se trata de un algoritmo compuesto por una red SP (Sustitución-Permutación) de 31 rondas. El bloque es de longitud 64 bits, y admite dos tamaños distintos de clave; 80 y 128 bits.

Diagrama

Descripción generada automáticamente

1. Esquema de alto nivel de Present

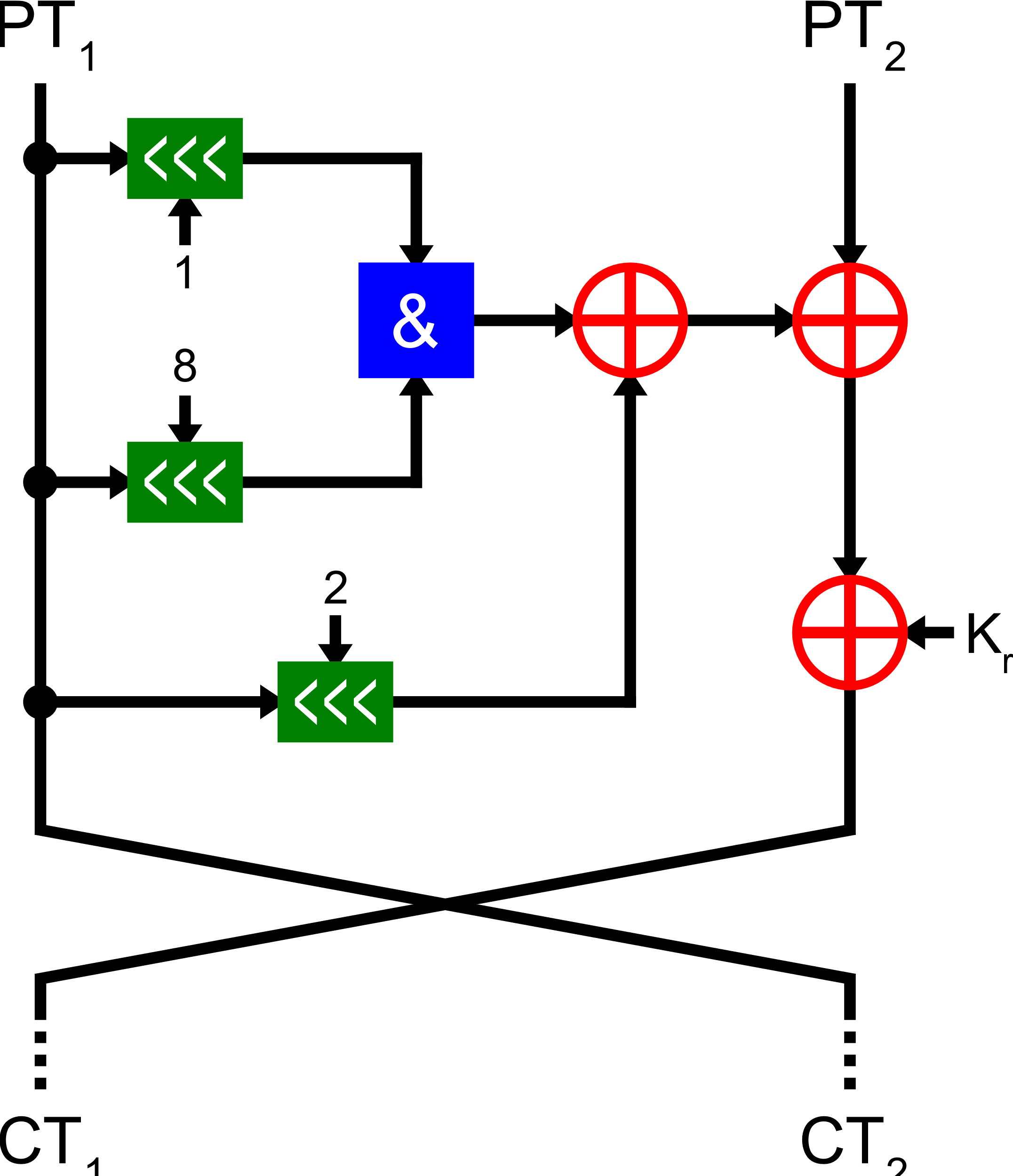
## SIMON

La familia de algoritmos de cifrado SIMON fue desarrollada y lanzada por la NSA en el año 2013. SIMON está optimizado para su uso en hardware, mientras que SPECK, del que hablaremos a continuación, está enfocado al software.

Estos cifrados comenzaron su desarrollo en 2011 con una serie de necesidades para dispositivos IoT en mente, y la NSA presionó mucho para intentar incluirlos en el standard internacional.

Esto se ralentizó ya que países como Alemania, Japón o Israel se opusieron, alegando que la NSA estaba tratando de estandarizarlos a sabiendas de sus debilidades, cosa que la NSA sigue negando hoy en día. A pesar de esto, en 2018 fueron aceptados como standard para RFID.

Este cifrado se basa en una red de Feistel con una palabra de *n* bits, por lo que su longitud de bloque es de *2n*. Su clave *m* tiene longitud múltiplo de 2, 3 o 4 por *n.* A la hora de referirnos a un cifrado de SIMON también se suele hacer como Simon*n/nm*. Así, por ejemplo, un cifrado Simon64/128 tendría una palabra de 32 bits y una clave de 128 bits.



2. Ronda de cifrado SIMON

## SPECK

Como hemos mencionado antes, el cifrado SPECK es la contraparte de SIMON, con un enfoque en el software.

En este caso, el algoritmo consta de un bloque formado siempre por dos palabras, que pueden variar su longitud en bits entre 16, 24, 32, 48 o 64. Por otra parte, su clave puede ser de 2, 3 o 4 palabras. Cada ronda consiste en dos rotaciones, sumándole la palabra derecha a la izquierda, realizando una operación XOR entre la clave y la palabra izquierda y otro XOR de la palabra izquierda con la palabra derecha.

El número de rondas depende de los parámetros seleccionados:

|  |  |  |
| --- | --- | --- |
| Tamaño del bloque en bits | Tamaño de la clave en bits | Rondas |
| 2 × 16 = 32 | 4 × 16 = 64 | 22 |
| 2 × 24 = 48 | 3 × 24 = 72 | 22 |
| 4 × 24 = 96 | 23 |
| 2 × 32 = 64 | 3 × 32 = 96 | 26 |
| 4 × 32 = 128 | 27 |
| 2 × 48 = 96 | 2 × 48 = 96 | 28 |
| 3 × 48 = 144 | 29 |
| 2 × 64 = 128 | 2 × 64 = 128 | 32 |
| 3 × 64 = 192 | 33 |
| 4 × 64= 256 | 34 |

Diagrama, Esquemático

Descripción generada automáticamente

3. Tres rondas de Speck con 2 palabras por clave

# Funciones Hash

Las funciones hash son algoritmos criptográficos que reciben un dato de entrada en una salida en forma de texto que, dependiendo de la función, puede tener tamaño variable o siempre fijo independientemente del tamaño de los datos recibidos.

Son comúnmente utilizadas en criptografía para almacenar contraseñas en bases de datos sin necesidad de guardarlas en limpio, y a continuación vamos a tratar dos de las funciones hash ligeras más comunes en IoT.

## PHOTON

Esta familia de funciones hash ligera fue diseñada por Jian Guo, Thoman Peyrin y Axel Poschmann como respuesta a la necesidad de una función hash ligera aplicable a etiquetas RFID.

El diseño de esta familia de funciones, al menos en el momento de su publicación, la convirtió en la función hash más compacta conocida hasta entonces.

Este algoritmo tiene una estructura inspirada en las funciones esponja (funciones criptográficas con un estado interno que pueden tomar como entrada un stream de bits de cualquier tamaño y producir una salida de un tamaño deseado), siendo su principal diferencia la capacidad de utilizar un bitrate distinto para la salida que el empleado en la entrada.

Diagrama, Esquemático

Descripción generada automáticamente

Por otra parte, sus permutaciones están muy inspiradas en la estructura de cifrado AES, donde el estado interno puede ser representado por una matriz matriz cuadrada de tamaño *d.d*, donde cada permutación interna se compone de 12 rondas.

Tabla

Descripción generada automáticamente con confianza media

## QUARK

La familia de funciones Quark fue desarrollada por Jean-Philippe Aumasson, Luca Henzen, Willi Meier y María Naya-Plasencia, y nació con la mente puesta en las etiquetas RFID.

Esta familia se basa en una construcción en esponja y un único nivel de seguridad con el fin de reducir las necesidades de memoria.

Está inspirada por otros protocolos ligeros como Grain o KATAN y se compone de tres instancias: u-Quark, d-Quark, and t-Quark.

Esta familia de funciones hash ha sido muy comparada con la familia anteriormente mencionada, PHOTON. En estas comparaciones se comprobó que si bien ambas familias de algoritmos son muy similares, Quark está menos optimizada para un uso a nivel de software, aunque esto no tiene demasiada importancia ya que su implementación está pensada para ser realizada a nivel de hardware. En conclusión, ambos tienen un buen equilibrio entre rendimiento y seguridad.

Cabe también destacar que, en el momento de dicha comparación, ninguno de los dos algoritmos ha sido roto aún.

# Cifrado en flujo

Los algoritmos de cifrado en flujo son cifrados de clave simétrica en los que la entrada pasa por un flujo de claves o *keystream* generada a partir de la clave de cifrado, en las que se opera dígito por dígito para obtener una salida codificada.

Estos algoritmos suelen tener el problema de depender de un generador de números pseudoaleatorios (PRNG) que cumpla unos mínimos de calidad en lo que a ser criptográficamente seguros se refiere.

## Grain

Este algoritmo de cifrado fue subido a eSTREAM en 2004, y está diseñado para funcionar en entornos de hardware restrictivos, lo que lo vuelve una buena opción para su aplicación en IoT.

A día de hoy, se le han encontrado númerosas vulnerabilidades, las cuales han sido en su mayoría corregidas en Grain 128a, que es la versión recomendada y que aporta seguridad en 128 bits y mecanismos de autenticación.