**DAVID RICARDO CRUZ JUAREZ**

ChaCha20-Poly1305 y AES-GCM (incluyendo AES-256-GCM) son dos esquemas de cifrado autenticado que se utilizan para proporcionar confidencialidad e integridad a los datos. A continuación, te describo las diferencias clave entre ambos y si tienen soporte en sistemas embebidos:

ChaCha20-Poly1305:

* Algoritmo de cifrado\*\*: ChaCha20 es un cifrado de flujo, lo que significa que opera byte a byte, mientras que Poly1305 es un código de autenticación basado en el cálculo de polinomios.
* Velocidad: ChaCha20 es generalmente más rápido que AES-GCM en la mayoría de las plataformas, lo que lo hace atractivo para sistemas con recursos limitados.
* \*\*Seguridad\*\*: Ambos son considerados seguros, pero algunos argumentan que ChaCha20-Poly1305 es más resistente a ciertos tipos de ataques en comparación con AES-GCM.

AES-GCM (incluyendo AES-256-GCM):

* Algoritmo de cifrado\*\*: Utiliza el cifrado AES (Advanced Encryption Standard), que es un cifrado de bloque.
* \*\*Velocidad\*\*: Aunque puede ser más lento que ChaCha20 en algunas implementaciones, AES-GCM sigue siendo lo suficientemente rápido para muchas aplicaciones y es ampliamente utilizado.
* \*\*Seguridad\*\*: AES es ampliamente aceptado y se ha sometido a pruebas exhaustivas, y AES-GCM es considerado muy seguro.

En cuanto al soporte en sistemas embebidos, tanto ChaCha20-Poly1305 como AES-GCM son algoritmos de cifrado y autenticación utilizados en sistemas embebidos, pero la disponibilidad puede depender de la plataforma y la implementación específica. Ambos algoritmos son eficientes y se utilizan en una variedad de sistemas, desde dispositivos IoT hasta sistemas integrados en hardware dedicado.

Es importante verificar si la biblioteca de criptografía que se utiliza en un sistema embebido en particular ofrece soporte para ChaCha20-Poly1305 y/o AES-GCM, y si cumplen con los requisitos de seguridad y rendimiento específicos de tu aplicación. La elección entre los dos dependerá de tus necesidades particulares y las limitaciones de recursos del sistema.

LINK de Repositorio

<https://replit.com/join/gnwrawxbmb-davidrcruzjuare>

Codigo Fuente

#include <stdio.h>

#include "sodium.h"

#define MESSAGE (const unsigned char \*) "test"

#define MESSAGE\_LEN 4

#define ADDITIONAL\_DATA (const unsigned char \*) "123456"

#define ADDITIONAL\_DATA\_LEN 6

unsigned char nonce**[**crypto\_aead\_aes256gcm\_NPUBBYTES**];**

unsigned char key**[**crypto\_aead\_aes256gcm\_KEYBYTES**];**

unsigned char ciphertext**[**MESSAGE\_LEN **+** crypto\_aead\_aes256gcm\_ABYTES**];**

unsigned long long ciphertext\_len**;**

void print\_hex**(**const unsigned char **\***data**,** size\_t len**)** **{**

**for** **(**size\_t i **=** 0**;** i **<** len**;** i**++)** **{**

printf**(**"%02x "**,** data**[**i**]);**

**}**

printf**(**"\n"**);**

**}**

int main**(**void**)** **{**

**if** **(**sodium\_init**()** **<** 0**)** **{**

*// Puede que no se pueda inicializar Sodium*

**return** 1**;**

**}**

*// Definir la clave y el nonce*

randombytes\_buf**(**key**,** **sizeof** key**);**

randombytes\_buf**(**nonce**,** **sizeof** nonce**);**

*// Mensaje a cifrar*

const char **\***message **=** "test"**;**

const size\_t message\_len **=** strlen**(**message**);**

*// Cifrar el mensaje*

**if** **(**crypto\_aead\_aes256gcm\_encrypt**(**ciphertext**,** **&**ciphertext\_len**,**

**(**const unsigned char **\*)**message**,** message\_len**,**

ADDITIONAL\_DATA**,** ADDITIONAL\_DATA\_LEN**,** **NULL,** nonce**,** key**)** **!=** 0**)** **{**

printf**(**"Error al cifrar el mensaje.\n"**);**

**return** 1**;**

**}**

*// Imprimir el mensaje cifrado en formato hexadecimal*

printf**(**"Mensaje cifrado (hex):\n"**);**

print\_hex**(**ciphertext**,** ciphertext\_len**);**

**return** 0**;**

**}**

Preguntas

Porque en el compilador online me maraca estos erros 1 ves corre bien y de ahí siempre marca errores, como puedo solucionar esos errores

