**Esp32 – Flash Encryption**

No processo de criptografar a flash da esp32 uma chave AES-256 é grava no efuse da esp32, e após isso o firmware é gravado encriptografado na flash, e toda vez que for ser executado o código é descriptografado. Uma vez gravado a chave no efuse não é mais possível ler ou alterar seu conteúdo, portanto a chave estará em segurança. O código criptografado ainda é possível de ser lido da flash, mas a menos que tenhamos a chave não é possível descriptografar o seu conteúdo.

Os binários “Bootloader”, “Partition table”, “OTA DATA”, todas “APP (seu código)” e as partições marcadas com a flag “encrypted” na tabela de partição serão criptografados. Partições que não estiverem marcados com “encrypted” não serão criptografados e poderão ser lidos externamente, tenha atenção ao utilizar APIs para acesso da flash como NVS e SPIFFS.

**1 Criando a chave:**

python espsecure.py generate\_flash\_encryption\_key key.bin

**2 Gravando a chave:**

python espefuse.py --port /dev/ttyUSB0 burn\_key flash\_encryption key.bin

**3 Ativando criptografia da flash:**

Menuconfig -> Security Features -> Enable Flash encryption on boot

(Como a chave já foi gravada a mesma será usada, caso contrário uma chave aleatória será gravada)

**4 Fazer o upload de forma padrão(plaintext)**

Este primeiro upload é em plain text, e o bootlaoder irá criptografar todo o conteúdo da memória e reiniciar, após reiniciar a flash estará criptografada. A partir de agora o eFuse FLASH\_CRYPT\_CNT deverá estar em 1, indicando que só será aceito uploads criptografados.

**5 Protegendo o eFuse FLASH\_CRYPT\_CNT**

Agora devemos proteger este eFuse contra escrita para impossibilitar quelaquer upload sem o conhecimento da chave, isso protege a placa de uploads não permitidos, aceitando hardwares que tenham sidos criptografados com a mesma chave gravada no efuse AES-256. Esta etapa já serve como uma forma de “Secure boot”.

Para isso vamos gravar o eFuse FLASH\_CRYPT\_CNT para um número ímpar.

python espefuse.py --port /dev/ttyUSB0 write\_protect\_efuse FLASH\_CRYPT\_CNT

Agora apenas uploads criptografados são permitidos.

**6 Fazendo o upload de binários criptografados**

*idf.py encrypted-flash monitor*

*Com este procedimento eu consegui bloquear qualquer upload não criptografado em minha placa pela entrada serial. Uploads não criptografados ainda são possíveis com OTA. Mas, ainda assim, o firmware é gravado criptografado após o término de atualização OTA. (testei na prática e funcionou)*

*Gostei dessa opção pois caso a chave seja perdida ainda é possível atualizar via OTA, e, como o servidor OTA é de nossa operação e opera em HTTPS o conteúdo do firmware ainda está seguro.*

Atualizações via serial ainda são possíveis, porém o firmware deve estar encriptografado com a mesma chave gravada no eFuse.

<https://www.embarcados.com.br/protecao-da-flash-no-esp32/>

Com a flash encryption a flash é criptografada com uma chave AES-256 e apenas é possível ler ela quem tem esta chave.

Secure boot garante que apenas um firmware criptografado com a chave correta pode ser bootado.