METODOLOGÍA DE DISEÑO PARA LA DETECCIÓN DE FALLOS EN CIFRADORES DE BLOQUE BASADA EN CÓDIGOS DE HAMMING

F. Eugenio Potestad-Ordóñez[1], Erica Tena-Sánchez[1], Pilar Parra-Fernández[1], Carmen Baena-Oliva1, Antonio J. Acosta-Jiménez[2], Manuel Valencia-Barrero[1] y Carlos Jesús Jiménez-Fernández[1]

[1]Departamento de Tecnología Electrónica, Universidad de Sevilla, Sevilla

[2]Departamento de Electrónica y Electromagnetismo, Universidad de Sevilla, Sevilla

E-mail de correspondencia: fpotestad@us.es

RESUMEN

La inserción de fallos y en concreto los análisis diferenciales de fallos (*Differential Fault Analysis – DFA*) se han convertido en uno de los principales métodos para explotar las vulnerabilidades de los cifradores de bloque utilizados en multitud de aplicaciones. En este trabajo se presenta un nuevo esquema de protección basado en generar firmas de los registros internos utilizando códigos de Hamming. Esto permite cubrir un gran número de tipos de fallos, detectando tanto cambios a nivel de bit pares e impares, así como cambios a nivel de byte, los cuales son los fallos explotables por los DFAs. Como caso de estudio, el esquema presentado se ha aplicado al cifrador de bloque estándar Advanced Encryption Standard (AES) implementado utilizando T-boxes. Los resultados obtenidos sugieren un alto nivel de cobertura de fallos con un coste de consumo de recursos del 16% y sin ninguna penalización en la degradación de frecuencia.

1. INTRODUCCIÓN

Aunque muchos de los estándares de cifrado existentes han demostrado ser matemáticamente seguros, existen técnicas que permiten atacar la implementación física de dichos algoritmos. En los ataques de análisis de fallos (*DFA*) el atacante intenta manipular el circuito de forma no permanente para generar errores de funcionamiento transitorios (fallos) y así obtener la información secreta que contiene el dispositivo.

En este trabajo tomamos como vehículo de prueba el cifrador de bloques estándar del NIST AES [1] basado en implementaciones de T-boxes. En la Figura 1 a) se muestra el proceso de encriptado estándar para una clave de 128 bits. En la Figura 1 b)

se representa de forma esquemática la implementación de este cifrador utilizando T-boxes en lugar de S-boxes.

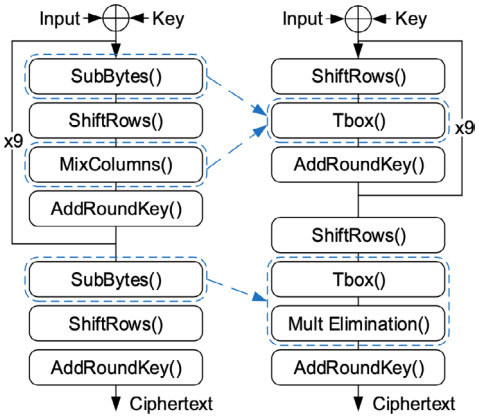

Figura 1. Representación esquemática de AES: a) Estándar, b) Basado en T-box.
Fuente: elaboración propia.

Los dos principales trabajos sobre DFA aplicados a AES son [2] y [3]. Estos trabajos establecen que si un atacante es capaz de inyectar diferentes tipos de fallos como son fallos de tipo único o múltiples bits, pares o impares y de tipo único o múltiples bytes, durante las últimas rondas de encriptado, el atacante podría recuperar la clave secreta.

Las vulnerabilidades reportadas del cifrado AES muestran que los fallos inyectados durante las operaciones SubBytes() y MixColumn() representan una fuga de información muy importante. Dado que estas operaciones se realizan a través de T-boxes, las vulnerabilidades reportadas son extensibles a este tipo de implementaciones con memorias y por lo tanto deben ser protegidas.

2. PROPUESTA DE ESQUEMA DE PROTECCIÓN

El esquema propuesto se basa en códigos de Hamming como generador de firmas de los datos utilizados por el cifrador. Aplicando este esquema para la detección de fallos en el cifrador AES basado en T-box, y utilizando Fórmula 1, es posible obtener una firma compuesta por 4 bits ($M_{0\text{-}3}$) para proteger 8 bits de datos ($D_{0\text{-}7}$). $M_{0\text{-}3}$ denota la firma de 4 bits añadida a los 8 bits de los datos procesados.

Fórmula 1. Operaciones para generación de firma.

$$M_0 = D_0 \oplus D_1 \oplus D_3 \oplus D_4 \oplus D_6$$
$$M_1 = D_0 \oplus D_2 \oplus D_3 \oplus D_5 \oplus D_6$$
$$M_2 = D_1 \oplus D_2 \oplus D_3 \oplus D_7$$
$$M_3 = D_4 \oplus D_5 \oplus D_6 \oplus D_7$$

Fuente: elaboración propia.

Generando una firma del dato de entrada de la operación T-box (I. Signature) y sobre la salida de esta misma operación (O. Signature), es posible fusionar y comprobar la firma final de unión (F. Signature) mediante una operación XOR como se muestra en Fórmula 2:

Fórmula 2. Obtención de la firma final.

$$F.\,Signature = I.\,Signature \oplus O.\,Signature$$

Fuente: elaboración propia.

Con este enfoque sólo es necesario almacenar y probar un único valor de 4 bits para cada byte que se quiera proteger. Este es el único valor adicional que se debe almacenar, como se muestra en la Figura 2.

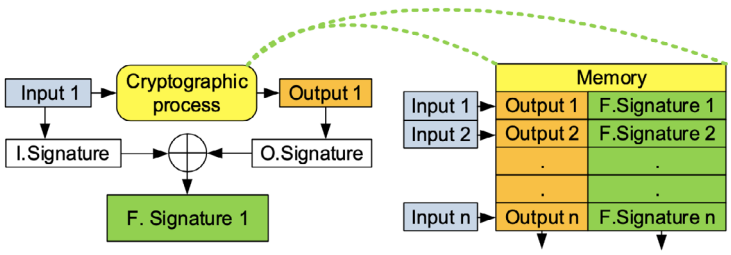

Figura 2. Representación del cálculo de la firma y utilización de la memoria.
Fuente: elaboración propia.

3. RESULTADOS

Se han realizado diferentes pruebas de simulación de fallos en las que se han considerado fallos pares/impares y de uno o varios bytes. Todos los fallos inyectados fueron detectados por nuestro esquema.

Los resultados muestran un coste de 645 Slice Registers y 720 LUTs para el cifrador protegido con la propuesta. Esto representa un aumento total del uso de recursos de un 25,5% más de Slice Registers y un 16% más de LUTs, y ningún uso adicional de BRAM. Esto se consigue sin degradación de la frecuencia.

Tabla 1. Comparación con diferentes esquemas de protección.

Propuesta	Sobrecoste en área	Degradación de frecuencia	Tipo de fallo detectado				Tecnología
Desprotegido	**1**	**1**	**No**	**No**	**No**	**No**	**Spartan 6**
[4]	1.08	0.70	Si	No	No	No	Virtex 1000
[5]	1.77	0.86	Si	Si	Si	Si	Virtex E
[6]	1.25	0.88	Si	Si	Si	Si	Virtex 5
Propuesta	1.16	1	Si	Si	Si	Si	Spartan 6

Fuente: elaboración propia.

En la Tabla 1 se muestra una comparativa con esquemas de protección previos. Esta tabla muestra la sobrecarga de área, la degradación de la frecuencia y el tipo de fallos detectados para las implementaciones de AES.

La solución [4] presenta el menor impacto en área, pero tiene cobertura de fallos pequeña y una alta penalización en frecuencia. Las soluciones [5] y [6], proporcionan una buena cobertura de fallos, pero con un mayor uso de recursos y con una degradación del throughput.

4. CONCLUSIONES

Los resultados obtenidos sugieren que la solución propuesta puede aplicarse con una sobrecarga de LUT de alrededor del 16% y sin ninguna penalización en frecuencia. La evaluación de la cobertura de fallos indica que el esquema propuesto es capaz de detectar todos los tipos de fallos de los DFAs: bits defectuosos pares e impares en el mismo o en diferentes bytes. Por último, la comparación con los esquemas previos sugiere que la solución propuesta es capaz de detectar todos los tipos de fallos en el mismo o en diferentes bytes con un coste de recursos significativamente menor, y sin degradar la frecuencia.

AGRADECIMIENTOS

Este trabajo ha sido financiado por Proyecto PID2020-116664RB-I00 financiado por MCIN/AEI/ 10.13039/501100011033, por Programa Operativo FEDER 2014-2020 and Consejería de Economía, Conocimiento, Empresas y Universidad de la Junta de Andalucía under Project US- 1380823 y por SPIRS (Secure Platform for ICT Systems Rooted at the Silicon Manfacturing Process) Project with Grant Agreement No. 952622 under the European Union's Horizon 2020 research and innovation programme.

REFERENCIAS BIBLIOGRÁFICAS

[1] **Daemen, J., & Rijmen, V.** (1999). AES proposal: Rijndael.

[2] **Giraud, C.** (2004, May). Dfa on aes. In International Conference on Advanced Encryption Standard (pp. 27-41). Springer, Berlin, Heidelberg.

[3] **Dusart, P., Letourneux, G., & Vivolo, O.** (2003, October). Differential fault analysis on AES. In International Conference on Applied Cryptography and Network Security (pp. 293-306). Springer, Berlin, Heidelberg.

[4] **Wu, K., Karri, R., Kuznetsov, G., & Goessel, M.** (2004, October). Low cost concurrent error detection for the advanced encryption standard. In 2004 International Conferce on Test (pp. 1242-1248). IEEE.

[5] **Karpovsky, M., Kulikowski, K. J., & Taubin, A.** (2004, June). Robust protection against fault-injection attacks on smart cards implementing the advanced encryption standard. In International Conference on Dependable Systems and Networks, 2004 (pp. 93-101). IEEE.

[6] **Mestiri, H., Benhadjyoussef, N., Machhout, M., & Tourki, R.** (2013). High performance and reliable fault detection scheme for the advanced encryption standard. International Review on Computers & Software (IRECOS), 8(3), 730-746.