Fault Based Almost Universal Forgeries on CLOC and SILC

<u>Avik Chakraborti</u> (ISI, Kolkata) Joint Work With Debapriya Basu Roy (IIT Kharagpur) Donghoon Chang (IIIT, Delhi) S V Dilip Kumar (IIT Kharagpur) Debdeep Mukhopadhyay (IIT Kharagpur) and Mridul Nandi (ISI, Kolkata)

September, 2016

Fault Analysis on CLOC and SILC

< □ > < 同 > < 三

Motivation

Description of CLOC and SILC Fault Based Almost Universal Forgery on CLOC Fault Based Almost Universal Forgery on SILC Implementation of Fault Conclusion



- 2 Description of CLOC and SILC
- 3 Fault Based Almost Universal Forgery on CLOC
- 4 Fault Based Almost Universal Forgery on SILC
- Implementation of Fault
- 6 Conclusion

< A >

- ₹ 🖬 🕨

Generic Fault Based Existential Forgery on AE Schemes

- Make a fault injected encryption query (N, A, M) and receive (C, T).
- Fault is injected at known bit positions *N* and *A* to result in *N*' and *A*' respectively.
- Make a valid forge with (N', A', C, T).

Non-Trivial

 $k~(k\gg1)$ forgery using one or very few faults

< D > < A > < B >

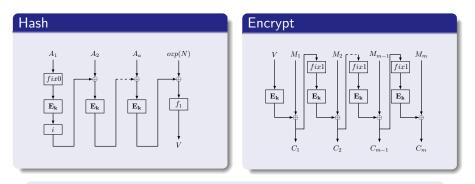


- 2 Description of CLOC and SILC
- 3 Fault Based Almost Universal Forgery on CLOC
- 4 Fault Based Almost Universal Forgery on SILC
- Implementation of Fault
- 6 Conclusion

< A >

.⊒ . ►

Description of CLOC



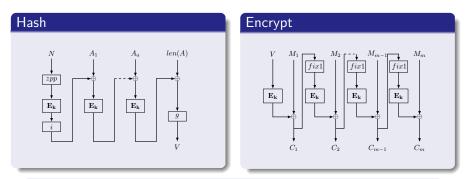
 $V \leftarrow Hash_{K}(N, A), C \leftarrow Enc_{K}(V, M), T \leftarrow PRF_{K}(V, C)$

・ロト ・回ト ・ヨト ・ヨト

æ

Description of SILC

Differes with CLOC in $Hash_K$. Enc_K and PRF_K are same.



 $V \leftarrow Hash_{K}(N, A), C \leftarrow Enc_{K}(V, M), T \leftarrow PRF_{K}(V, C)$

Fault Analysis on CLOC and SILC

Single Bit Fault Based Forgery on CLOC Almost Universal Fault Based Forgery on CLOC

Motivation

- 2 Description of CLOC and SILC
- Fault Based Almost Universal Forgery on CLOC
 Single Bit Fault Based Forgery on CLOC
 Almost Universal Fault Based Forgery on CLOC
- 4 Fault Based Almost Universal Forgery on SILC
- Implementation of Fault



< A >

Single Bit Fault Based Forgery on CLOC Almost Universal Fault Based Forgery on CLOC

Fault Model

Fault *e* injected at the first bit of the *n*-bit input state of the second block cipher call in Enc_{K} .

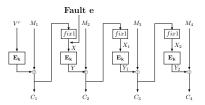


Image: A mathematical states and a mathem

Single Bit Fault Based Forgery on CLOC Almost Universal Fault Based Forgery on CLOC

Phase 1 of the Forgery

Construct a faulty ip/op pair and 2 valid ip/op pairs corresponding to E_K by one enc query.

1 enc query
$$(N^r, A^r, M = (M_1, M_2, M_3, M_4))$$

Receives $(C = (C_1, C_2, C_3, C_4), T)$

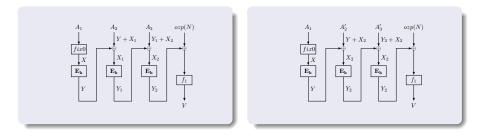
Computes $(X, Y), (X_1, Y_1), (X_2, Y_2)$

< ロ > < 同 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ >

Single Bit Fault Based Forgery on CLOC Almost Universal Fault Based Forgery on CLOC

Phase 2

Construct two colliding associated data (A, A'), that produces same V under same N



Fault Analysis on CLOC and SILC

(日) (同) (三) (三)

э

Single Bit Fault Based Forgery on CLOC Almost Universal Fault Based Forgery on CLOC

Phase 3 and Phase 4

Phase 3

• Construct (C^{*}, T^{*}) under N, A and M^{*} by a single encryption query

Phase 4

Fault Analysis on CLOC and SILC

(日)

э

Single Bit Fault Based Forgery on CLOC Almost Universal Fault Based Forgery on CLOC

Motivation

- 2 Description of CLOC and SILC
- Fault Based Almost Universal Forgery on CLOC
 Single Bit Fault Based Forgery on CLOC
 Almost Universal Fault Based Forgery on CLOC
- 4 Fault Based Almost Universal Forgery on SILC
- 5 Implementation of Fault



< A >

Single Bit Fault Based Forgery on CLOC Almost Universal Fault Based Forgery on CLOC

Different Steps for the Almost Universal Forgery on CLOC

Any
$$(N,A=(A_1,\cdots,A_a),M=(M_1,\cdots,M_m))$$
, except A_1 fixed

• Obtain faulty ip-op pair X and Y (like Phase 1)

•
$$A_1 = X$$

- Compute all BC ip-op pairs during A processing
- Requires *a* enc queries
- Find A' colliding with A at V
- Enc query: $(N, A', M) \rightarrow (C, T)$
- Forge with (N, A, C, T)

< D > < A > < B >

Single Bit Fault Based Forgery on CLOC Almost Universal Fault Based Forgery on CLOC

What does Almost Mean?

•
$$I_1 = A_1 = X, O_1 = Y = E_k(I_1)$$

•
$$X_1 = A_2 \oplus O_1, Y_1 = E_k(X_1)$$

•
$$X_{a-1} = A_a \oplus Y_{a-2}, Y_{a-1} = E_k(X_{a-1})$$

Restriction

• Only
$$A_1 = X$$

• No restrictions on N and M

Fault Analysis on CLOC and SILC

(日) (同) (三) (三)

э

Single Bit Fault Based Forgery on CLOC Almost Universal Fault Based Forgery on CLOC

First Encrytion Query

Query with N, A and any a single block message M^r = M^r₁.
Receive (C^r₁, T^r)

Compute $E_k(V) = M_1^r \oplus C_1^r$

Fault Analysis on CLOC and SILC

Single Bit Fault Based Forgery on CLOC Almost Universal Fault Based Forgery on CLOC

Next a-2 Encrytion Queries

For i=1 to a-2

Make an encryption query (N, A, M = (M'₁ = E_k(V) ⊕ X_i, M'₂) and receive (C' = (C'₁, C'₂), T').

• Compute
$$Y_i = M_2' \oplus C_2'$$

• Compute
$$X_{i+1} = A_{i+2} \oplus Y_i$$
.

<ロト < 同ト < 三ト

Single Bit Fault Based Forgery on CLOC Almost Universal Fault Based Forgery on CLOC

Last 2 Encrytion Queries

• Make an encryption query $(N, A, M = (M'_1 = E_k(V) \oplus X_{a-1}, M'_2)$ and receive $(C' = (C'_1, C'_2), T')$

• Compute
$$Y_{a-1} = M_2' \oplus C_2'$$

- Find a colliding associated data A' for A (colliding at V) (Same as Phase 2)
- Make an encryption query (N, A', M) and receive (C, T)

(日) (同) (三) (三)

Single Bit Fault Based Forgery on CLOC Almost Universal Fault Based Forgery on CLOC

Valid Forge

(N, A, C, T) is a Valid forge

Fault Analysis on CLOC and SILC

(日) (同) (三) (三)

э

Single Bit Fault Based Forgery on SILC Almost Universal Fault Based Forgery on SILC

Motivation

- Description of CLOC and SILC
- In the second second
- Fault Based Almost Universal Forgery on SILC
 Single Bit Fault Based Forgery on SILC
 Almost Universal Fault Based Forgery on SILC
- 5 Implementation of Fault



< 67 ▶

Single Bit Fault Based Forgery on SILC Almost Universal Fault Based Forgery on SILC

Motivation

- 2 Description of CLOC and SILC
- In the second second
- Fault Based Almost Universal Forgery on SILC
 Single Bit Fault Based Forgery on SILC
 Almost Universal Fault Based Forgery on SILC
- 5 Implementation of Fault



< 67 ▶

Single Bit Fault Based Forgery on SILC Almost Universal Fault Based Forgery on SILC

Fault Model

- Fault *e* injected at the first bit of the *n*-bit input state of the second block cipher call in *Enc_K*.
- Same as that of CLOC

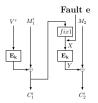
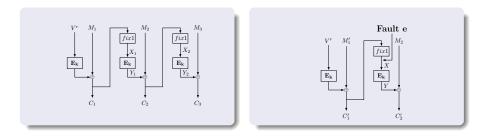


Image: A math a math

Single Bit Fault Based Forgery on SILC Almost Universal Fault Based Forgery on SILC

Phase 1 of the Forgery

Construct a *faulty* ip/op pair and 2 valid ip/op pairs to E_K by 2 enc queries.

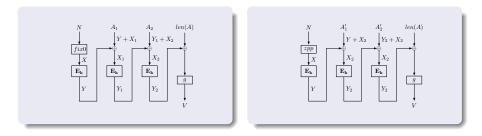


< □ > < 同 > < 回 >

Single Bit Fault Based Forgery on SILC Almost Universal Fault Based Forgery on SILC

Phase 2

Construct two colliding associated data (A, A'), that produces same V under same N



Fault Analysis on CLOC and SILC

(日) (同) (三) (三)

э

Single Bit Fault Based Forgery on SILC Almost Universal Fault Based Forgery on SILC

Phase 3 and Phase 4

Phase 3

• Construct (C^{*}, T^{*}) under N, A and M^{*} by a single encryption query

Phase 4

Fault Analysis on CLOC and SILC

<ロト < 同ト < 三ト

Single Bit Fault Based Forgery on SILC Almost Universal Fault Based Forgery on SILC

Motivation

- 2 Description of CLOC and SILC
- In the second second
- Fault Based Almost Universal Forgery on SILC
 Single Bit Fault Based Forgery on SILC
 Almost Universal Fault Based Forgery on SILC
- 5 Implementation of Fault



< 67 ▶

Single Bit Fault Based Forgery on SILC Almost Universal Fault Based Forgery on SILC

Different Steps for Almost Universal Forgery

Any (N, A, M), except N fixed, first bit of A_i , $1 \le i \le a$ is restricted

- Obtain faulty ip-op pair X and Y (like Phase 1)
- zpp(N) = X
- Compute all BC ip-op pairs during A processing
- Requires a + 1 enc queries
- Find A' colliding with A at V
- Enc query: $(N, A', M) \rightarrow (C, T)$
- Forge with (N, A, C, T)

< D > < A > < B >

Single Bit Fault Based Forgery on SILC Almost Universal Fault Based Forgery on SILC

What does Almost Mean?

•
$$X_1 = zpp(N) = X, Y_1 = Y = E_k(X_1)$$

•
$$X_2 = A_1 \oplus (Y_1), Y_2 = E_k(X_2)$$

•
$$X_{a+1} = A_a \oplus Y_a, Y_{a+1} = E_k(X_{a+1})$$

Restriction

•
$$zpp(N) = X$$
 and $X_1 = Y \oplus A_1$

• No restrictions on M

The rest of the attack is same

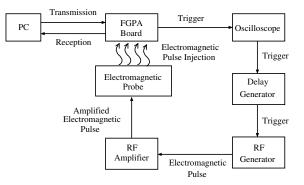


- 2 Description of CLOC and SILC
- 3 Fault Based Almost Universal Forgery on CLOC
- 4 Fault Based Almost Universal Forgery on SILC
- 5 Implementation of Fault
- 6 Conclusion

< A >

- ∢ ≣ ▶

Fault Attack Setup



Fault Analysis on CLOC and SILC

э

Implementation Results

- Implemented in SPARTAN-6 FPGA of SAKURA-G board
- LUT 1000, Registers 1000, Slices 1000, Critical path 6ns
- Focus only on fix1 module, fix1 module have been ported
- 32 bit left shift in the output of fix1 module
- Input a random *M* with 95th bit 0 and inject fault
- After fault First bit of *M* is 0



- 2 Description of CLOC and SILC
- 3 Fault Based Almost Universal Forgery on CLOC
- 4 Fault Based Almost Universal Forgery on SILC
- 5 Implementation of Fault



< A >

- ₹ 🖬 🕨



- Fault based Almost Universal forgery on CLOC
- Fault based Almost Universal forgery on SILC Implementation of Fault

Thank you

Fault Analysis on CLOC and SILC

< □ > < 同 > < 回 >