Fault Based Almost Universal Forgeries on CLOC and SILC

Avik Chakraborti (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

Generic Fault Based Existential Forgery on AE Schemes

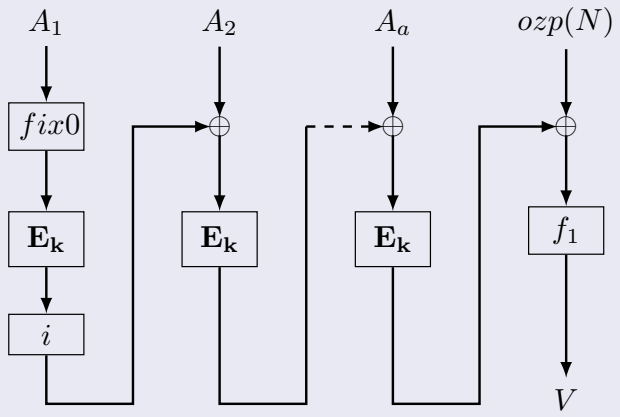
- Make a fault injected encryption query (N, A, M) and receive (C, T).
- Fault is injected at known bit positions N and A to result in N' and A' respectively.
- Make a valid forge with (N', A', C, T).

Non-Trivial

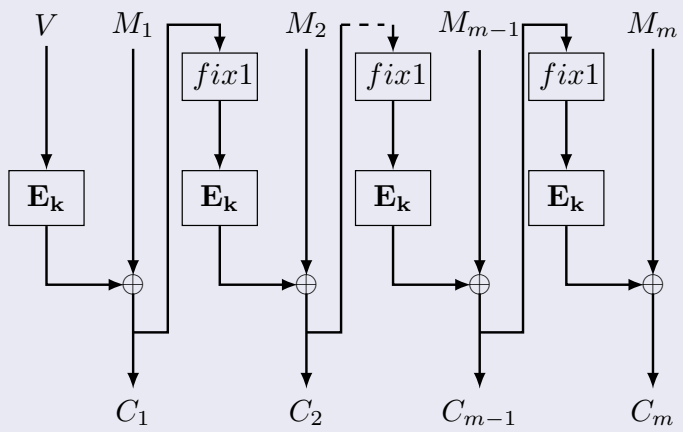
k $(k \gg 1)$ forgery using one or very few faults

Description of CLOC

Hash



Encrypt

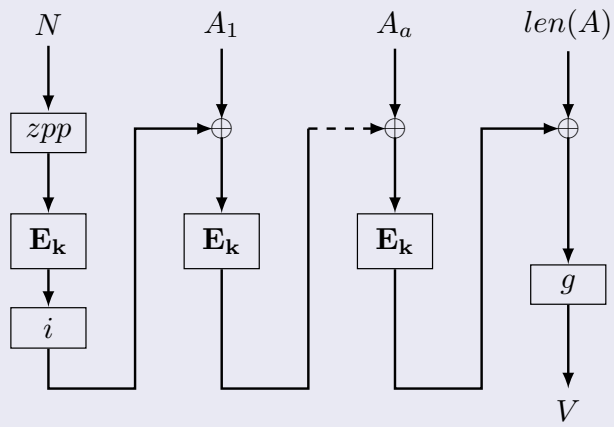


$$V \leftarrow Hash_K(N, A), C \leftarrow Enc_K(V, M), T \leftarrow PRF_K(V, C)$$

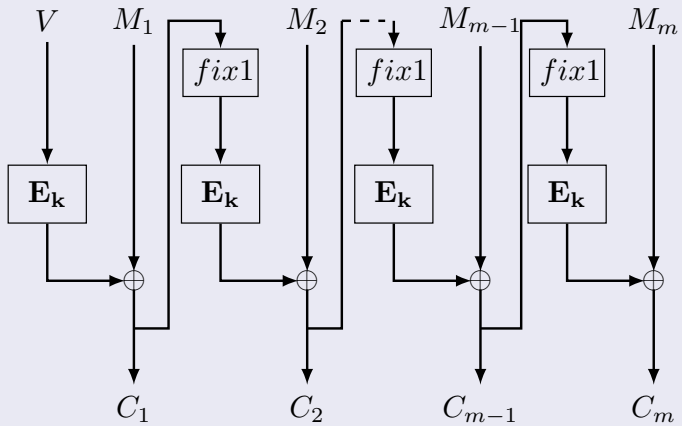
Description of SILC

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

Hash



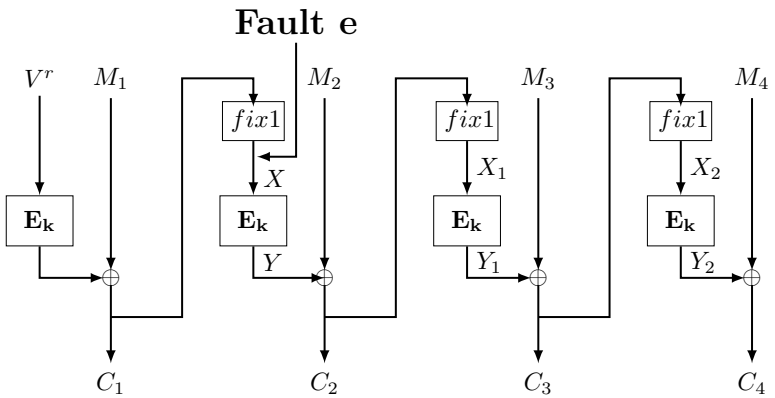
Encrypt



$$V \leftarrow Hash_K(N, A), C \leftarrow Enc_K(V, M), T \leftarrow PRF_K(V, C)$$

Fault Model

Fault e injected at the first bit of the n-bit input state of the second block cipher call in Enc_K.



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.

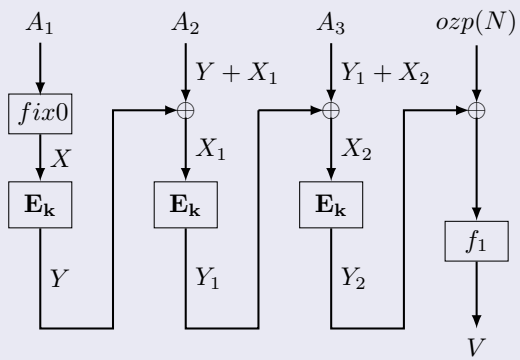
1 enc query $(N^r, A^r, M = (M_1, M_2, M_3, M_4))$

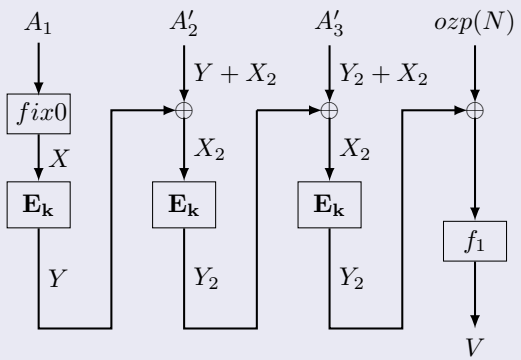
Receives $(C = (C_1, C_2, C_3, C_4), T)$

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

Phase 2

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





Phase 3 and Phase 4

Phase 3

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

Phase 4

- Forge (N, A', C^*, T^*)

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)

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

First Encrytion Query

- Query with N, A and any a single block message $M^r = M_1^r$.
- Receive (C_1^r, T^r)

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

Next a-2 Encrytion Queries

For i=1 to a-2

- Make an encryption query $(N, A, M = (M_1' = E_k(V) \oplus X_i, M_2')$ and receive $(C' = (C_1', C_2'), T')$.
- Compute $Y_i = M_2' \oplus C_2'$.
- Compute $X_{i+1} = A_{i+2} \oplus Y_i$.

Last 2 Encrytion Queries

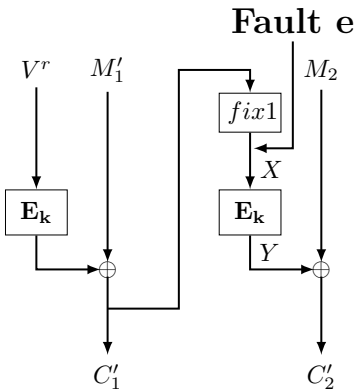
- Make an encryption query $(N, A, M = (M'_1 = E_k(V) \oplus X_{a-1}, M'_2)$ and receive $(C' = (C'_1, C'_2), T')$
- Compute $Y_{a-1} = M'_2 \oplus C'_2$
- Find a colliding associated data A' for A (colliding at V) (Same as Phase 2)
- Make an encryption query (N, A', M) and receive (C, T)

Valid Forge

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

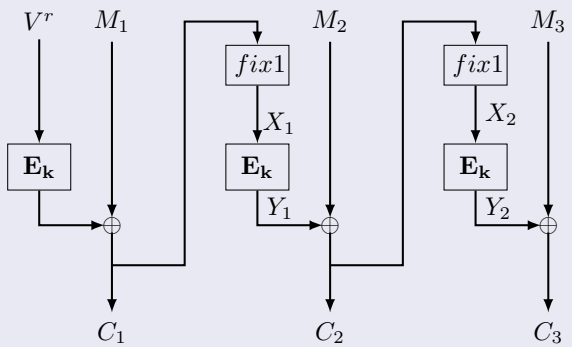
Fault Model

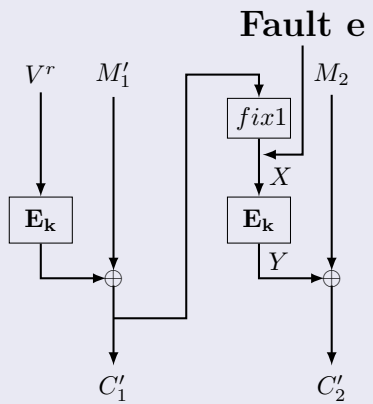
- Fault e injected at the first bit of the n-bit input state of the second block cipher call in Enc_K.
- Same as that of CLOC



Phase 1 of the Forgery

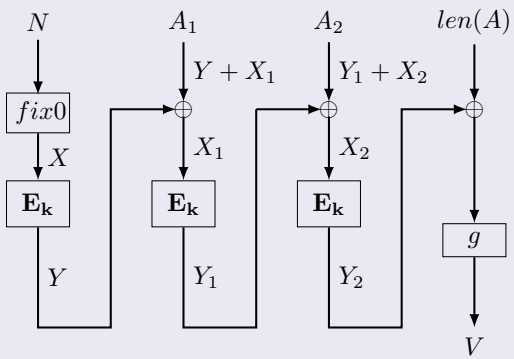
Construct a *faulty* ip/op pair and 2 valid ip/op pairs to E_K by 2 enc queries.

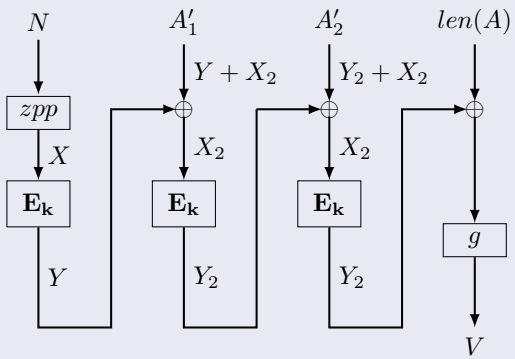




Phase 2

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





Phase 3 and Phase 4

Phase 3

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

Phase 4

- Forge (N, A', C^*, T^*)

Different Steps for Almost Universal Forgery

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

What does Almost Mean?

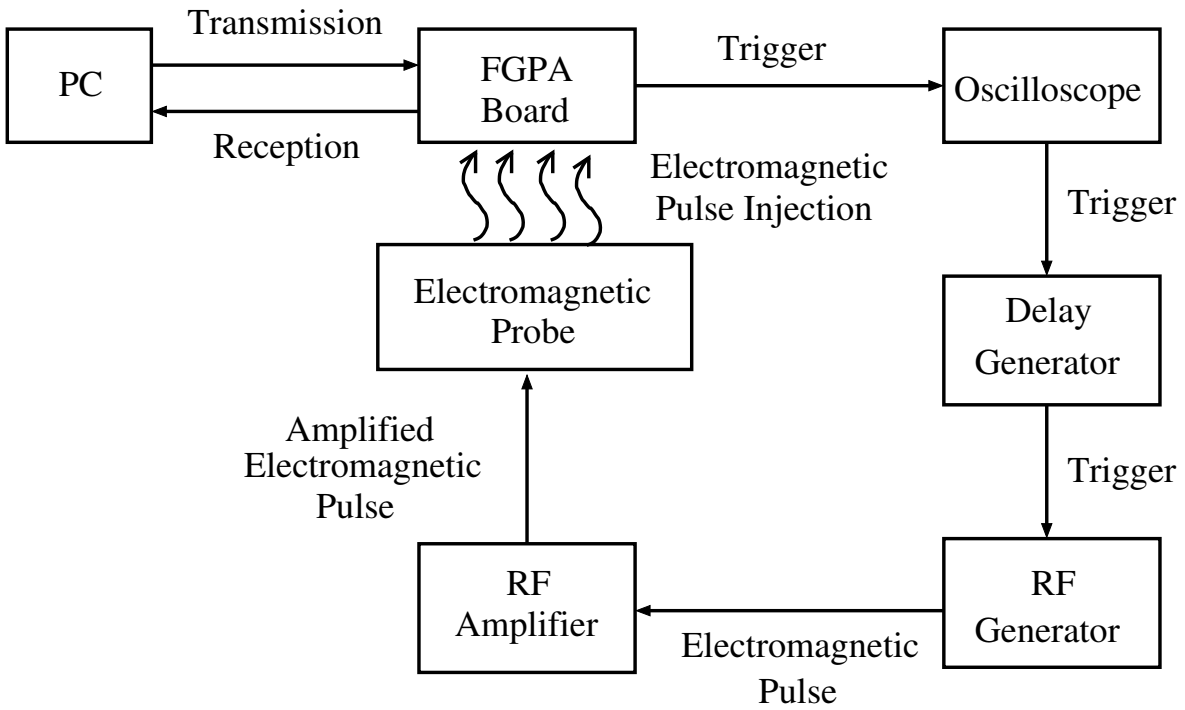
- $X_1 = zpp(N) = X, Y_1 = Y = E_k(X_1)$
- $X_2 = A_1 \oplus (Y_1), Y_2 = E_k(X_2)$
- $X_{a+1} = A_a \oplus Y_a, Y_{a+1} = E_k(X_{a+1})$

Restriction

- $zpp(N) = X$ and $X_1 = Y \oplus A_1$
- No restrictions on M

The rest of the attack is same

Fault Attack Setup



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 95^{th} bit 0 and inject fault
- After fault - First bit of M is 0

Conclusion

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

Thank you