An Overview of Authenticated Encryption

Mridul Nandi

Indian Statistical Institute, Kolkata

21 SEPTEMBER 2016
1 Introduction to Authenticated Encryption.

2 Security Notion of Authenticated Encryption

3 Mounting INT-RUP Attack on Popular AE Schemes

4 INT-RUP Analysis for “High-rate” Affine Constructions

5 Another Fault Based Almost Universal Forgery on CLOC-SILC
Introduction to Authenticated Encryption.

Security Notion of Authenticated Encryption
Mounting INT-RUP Attack on Popular AE Schemes
INT-RUP Analysis for “High-rate” Affine Constructions
Another Fault Based Almost Universal Forgery on CLOC-SILC

Encryption and Authentication

We need Encryption

We need MAC

Eve hears $M$

Alice knows message $M$

Alice sends $M$ to Bob

Mallet intercepts $M$ and sends $M'$ to Bob.

May or may not be the same as $M'$

Can Bob detect whether $M' = M$?
**Introduction to Authenticated Encryption.**

**Security Notion of Authenticated Encryption**

**Mounting INT-RUP Attack on Popular AE Schemes**

**INT-RUP Analysis for “High-rate” Affine Constructions**

**Another Fault Based Almost Universal Forgery on CLOC-SILC**

---

**Encryption and Authentication**

---

<table>
<thead>
<tr>
<th>Formally..</th>
</tr>
</thead>
<tbody>
<tr>
<td>$E : \mathcal{N} \times \mathcal{M} \times \mathcal{K} \rightarrow \mathcal{N} \times \mathcal{C}$</td>
</tr>
<tr>
<td>$D : \mathcal{N} \times \mathcal{C} \times \mathcal{K} \rightarrow \mathcal{M}$</td>
</tr>
<tr>
<td>Correctness Condition: $D(E(N, M, K), K) = M$</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Formally..</th>
</tr>
</thead>
<tbody>
<tr>
<td>$MAC : \mathcal{N} \times \mathcal{M} \times \mathcal{K} \rightarrow \mathcal{N} \times \mathcal{T}$</td>
</tr>
<tr>
<td>$\nu : \mathcal{N} \times \mathcal{T} \times \mathcal{M} \times \mathcal{K} \rightarrow {1, \bot}$</td>
</tr>
<tr>
<td>Correctness Condition: $\nu(MAC(N, M, T, K), M, K) = 1$</td>
</tr>
</tbody>
</table>
Why AE?

In practise *both Privacy* and *Authenticity* are desirable.

**Example**: A doctor wishes to send medical information $M$ about Alice to the medical database. Then

- We want data *privacy* to ensure Alice’s medical records remain *confidential*.
- We want *integrity* to ensure the person sending the information is really the doctor and the information was not modified in transit.

We need *authenticated encryption*.
Authenticated Encryption (AE)

Formally....

- \( AE.\mathcal{E} : M \times D \times N \times K \rightarrow C \)
- \( AE.\mathcal{D} : C \times D \times N \times K \rightarrow M \cup \perp \)

<table>
<thead>
<tr>
<th>Goal</th>
<th>Primitive</th>
<th>Security</th>
</tr>
</thead>
<tbody>
<tr>
<td>Privacy</td>
<td>Symmetric Encryption</td>
<td>IND-CCA/CPA</td>
</tr>
<tr>
<td>Integrity</td>
<td>MAC</td>
<td>UF-CMA</td>
</tr>
</tbody>
</table>

Table: Security Properties
Authenticated Encryption (AE)

\( \mathcal{M}, \mathcal{C}, \mathcal{D}, \mathcal{N}, \mathcal{K} \)

- \( \mathcal{M} \) - Message Space
- \( \mathcal{C} \) - Ciphertext Space
- \( \mathcal{K} \) - Key Space
- \( \mathcal{N} \) - Nonce Space
- \( \mathcal{A} \) - Associated Data Space

**Nonce**

- Arbitrary number used only *once*.
- Useful as initialization vectors. Example: *Counter*.

**Associated Data**

- *Header* of the Message
- Example: *IP Address*. 
1. Introduction to Authenticated Encryption.

2. Security Notion of Authenticated Encryption
   - Privacy and Authenticity of AE.
   - AE under INT-RUP Model.


4. INT-RUP Analysis for “High-rate” Affine Constructions

5. Another Fault Based Almost Universal Forgery on CLOC-SILC
Security of Authenticated Encryption.

Privacy

We want *IND-CPA*.

Integrity

- **Adversary’s Goal**: Receiver Accepts a “non-authentic” and *valid* ciphertext *C*.
- **INT-CTX**: *C* is “non-authentic” if it was never transmitted by the sender.

Goal - *IND-CPA + INT-CTX*. 
IND-CPA Security for Privacy

\[ \Delta_A(O_1; O_2) = | \Pr[A^{O_1} = 1] - \Pr[A^{O_2} = 1] |. \]

- \( \text{Adv}_{\text{AE}}^{\text{PRIV}} (A) := \Delta_A(\mathcal{E}_K; \$) \)
- \( \text{Adv}_{\text{AE}}^{\text{PRIV}} (q, \sigma, t) = \max_A \text{Adv}_{\text{AE}}^{\text{PRIV}} (A) \)
- \( t: \text{Time, } q: \# \text{queries, } \sigma: \# \text{ blocks in all queries} \)
INT-CTXT Security for Integrity

- $A$ forges if $\exists (N_j^*, A_j^*, C_j^*, T_j^*) \ni \nu_k(N_j^*, A_j^*, C_j^*, T_j^*) = 1$

$\text{Adv}_{\mathcal{AE}}^{\text{INT}}(A) := \Pr[A \mathcal{E}_k \text{ forges}]$

$\text{Adv}_{\mathcal{AE}}^{\text{INT}}((q_e, q_f), (\sigma_e, \sigma_f), t) = \max_A \text{Adv}_{\mathcal{AE}}^{\text{INT}}(A)$
Issues on AE: Limited Buffer Implementation

- Implementation in small devices like smart card, which has limited buffer.

- **Limited buffer** $\Rightarrow$ **Release of unverified plaintext**
  (If Decryption query length is more than buffer size)
INT-RUP Model

• Proposed by Andreeva et.al. (Asiacrypt 2014)

• Integrity in RUP: Adversary can make decryption calls as well.

• Privacy in RUP: Similar to SPRP security.
INT-RUP Security for Integrity

\[ \text{Adv}^{\text{INT-RUP}}_{\text{AE}}(A) := \text{Pr}[A^{E_k, D_k} \text{ forges}] \]
Introduction to Authenticated Encryption.

Security Notion of Authenticated Encryption

Mounting INT-RUP Attack on Popular AE Schemes

INT-RUP Analysis for “High-rate” Affine Constructions

Another Fault Based Almost Universal Forgery on CLOC-SILC
If we choose randomly $M_0^1, M_1^1, \ldots, M_0^n, M_1^n$ then $\exists j_1, \ldots, j_n$, $M_1^{j_1} \oplus \cdots M_n^{j_n} = CS$ (target checksum) with high probability.

We can obtain all $M_i^{j_i}$'s making 2 unverified plaintext queries - $(C_1^0, \ldots, C_n^0)$ and $(C_1^1, \ldots, C_n^1)$.
Need final state matching as well as checksum matching.

Let $l = n + 1$. Final state $Y_{n+1}$ matching is easy as it depends only on the last ciphertext block $C_{n+1}$.
Integrity Attack on COPA under INT-RUP model

- Again, if we choose randomly $M_0^0, M_1^1, \ldots, M_n^0, M_n^1$ then $\exists j_1, \ldots, j_n, M_1^{j_1} \oplus \cdots \oplus M_n^{j_n} = CS$ with high probability.
- $M_i^{j_i}$ is obtained from $C_{i}^{j_i-1}$ and $C_{i}^{j_i}$.
We can obtain all $M_i^j$'s with same final state making 4 unverified plaintext queries - $(C_0^0, C_2, \ldots, C_n^0, C_n^*_{n+1})$, $(C_0^0, C_2^1, \ldots, C_n^0, C_n^*_{n+1})$, $(C_1^1, C_2^0, \ldots, C_n^1, C_n^*_{n+1})$ and $(C_1^1, C_2^1, \ldots, C_n^1, C_n^*)$. 
Find an internal state collision.

Use Attack approach like OCB, COPA (multi-collision type).
Finding Internal State Collision

Adaptive Adversarial Queries.

- \( C_{1..3}^1 \leftarrow E_K(M_1[1..3] = M_1^1 || * || *) \)
- \( M_{1..3}^2 \leftarrow D_K(* || C_2^1 || *) \)
- \( M_{1..3}^3 \leftarrow D_K(* || * || C_3^1) \)
- \( C_{1..3}^4 \leftarrow E_K(* || M_2^2 || M_3^3) \)

Main Observation

The final state \( W \) for the last enc query is independent of \( M_1^1 \).
Finding Internal State Collision

Why $W$ for the last enc query is independent of $M_1^1$?

- Contribution of $X_1^1$ to $W$ from both paths cancels out.

\begin{tikzpicture}
  
  \node (x1) at (0,0) {$X_1^1$};
  \node (y1) at (1,1) {$Y_1^1$};
  \node (x2) at (2,0) {$X_2^2$};
  \node (y2) at (1,-1) {$Y_2^1$};
  \node (x3) at (0,-2) {$X_3^3$};
  \node (y3) at (-1,-1) {$Y_3^1$};
  \node (w) at (1,-2) {$W$};

  \path[->] (x1) edge node [above] {3} (y1);
  \path[->] (y1) edge node [above] {1} (x2);
  \path[->] (x2) edge node [above] {2} (w);
  \path[->] (w) edge node [below] {1} (x3);
  \path[->] (x3) edge node [below] {3.2} (y3);
  \path[->] (y3) edge node [below] {1} (y2);
  \path[->] (y2) edge node [below] {3} (x1);
\end{tikzpicture}
Introduction to Authenticated Encryption

Security Notion of Authenticated Encryption

Mounting INT-RUP Attack on Popular AE Schemes

INT-RUP Analysis for “High-rate” Affine Constructions

Another Fault Based Almost Universal Forgery on CLOC-SILC

**Integrity Attack (NM) on ELmD under INT-RUP model**

**INT-RUP Attack on ELmD**

- With 8-many 3-block queries, we find a state collision.
- Extend State Collision attack to mount Integrity Attack

Mridul Nandi
Integrity Attack (NR) on ELMd under INT-RUP model

Approach (Similar)
- Find an internal state collision.
- Extend State Collision attack to mount Integrity Attack.

Non-triviality
- Nonce Respecting $\Rightarrow$ Single Enc Query.
- Previous Approach won’t work to find state collision.
Finding Internal State Collision (Demo example with $n = 4$)

Adaptive Adversarial Queries (Based on Primitive Polynomial)

(Assumption: Prim poly: $x^4 + x + 1$)

- $M^1_{1..5} \leftarrow D_K(C^1_{1..5})$
- $M^2_{1..5} \leftarrow D_K(C^2_1 \parallel C^1_{2..5})$
- $C^3_{1..5} \leftarrow E_K(M^1_1 \parallel M^2_{2..4} \parallel M^1_5)$

Main Observation

The final state $W$ for enc query matches with 2$^{nd}$ dec query.
Internal State Collision (Demo example with $n = 4$)

Finding Contributions of $Y$-variables in $W$

- Contribution of $Y_j^1$ ($j > 1$) in $W$ is 1 (Trivial).
- Contribution of $Y_1^1$ in $W$ is $2^4 + 3 = (2^4 + 2 + 1) = 0$.
- Contribution of $Y_1^2$ in $W$ is $3(2^3 + 2^2 + 2) = 1$. 

\[ Y_1^1 \rightarrow X_1^1 \rightarrow 2^4 \rightarrow W \]
\[ Y_1^2 \rightarrow X_2^2 \rightarrow 2^3 \rightarrow W \]
Integrity Attack (NR) on ELMd under INT-RUP model

Finding an internal collision

- Using same idea as the demo, one can find a state collision attack in $n + 1$-block messages.
- This works for any values of $n$ and for any primitive poly of degree $n$.

Extend State Collision attack to mount Integrity Attack

- Similar to the approach used in ELMd (NM).
Limited Buffer Implementation

- Masking plaintext by pseudorandom keystream. If valid release the seed. [Fouque et al. 2004]
- Ciphertext includes intermediate tag to verify part of ciphertext.
- Some other methods specific to constructions are also known e.g., delayed output for CBC-type constructions [Fouque et al. 2003] etc.
1. Introduction to Authenticated Encryption.

2. Security Notion of Authenticated Encryption


4. INT-RUP Analysis for “High-rate” Affine Constructions
   - A generic Attack on Rate-1 Affine AE
   - INT-RUP Analysis of CPFB (Rate 3/4)
   - mCPFB: A rate $\frac{3}{4}$ INT-RUP secure construction

5. Another Fault Based Almost Universal Forgery on CLOC-SILC
Rate-1 Affine Constructions

Rate of an AE Scheme

- Messages blocks processed per blockcipher call
- Rate-1 means one block processed per blockcipher call
- Efficient construction
Rate-1 Affine Constructions

\[ \begin{align*}
&\text{E[1,\ldots]} \\
&\pi_1 \rightarrow U_1 \\
&V_1 \\
&\pi_2 \rightarrow U_2 \\
&V_2 \\
&\vdots \\
&\pi_{I+1} \rightarrow U_{I+1} \\
&C \\
&\text{E[1+c+1,\ldots]} \\
&\pi_{I+c} \rightarrow V_{I+c} \\
&T \\
&\text{E[1+c+1,\ldots]} \\
\end{align*} \]
Rate-1 Affine Encryption

Encryption Matrix Representation

\[ E \cdot \begin{pmatrix} M \\ V^* = \begin{pmatrix} V \\ V_{tag} \end{pmatrix} \end{pmatrix} = \begin{pmatrix} U^* = \begin{pmatrix} U \\ U_{tag} \end{pmatrix} \\ Z = \begin{pmatrix} C \\ T \end{pmatrix} \end{pmatrix} \]

- \( E \): Enc Matrix, \( M \): Msg
- \( V \): Intermediate o/p from \( \pi \) during msg Process
- \( V_{tag} \): Intermediate o/p from \( \pi \) during tag Process
- \( U \): Intermediate i/p to \( \pi \) during M Process
- \( U_{tag} \): Intermediate i/p to \( \pi \) during tag Process
- \( C \): Ciphertext, \( T \): Tag

A generic Attack on Rate-1 Affine AE
INT-RUP Analysis of CPFB (Rate 3/4)
mCPFB: A rate \( \frac{3}{4} \) INT-RUP secure construction
Rate-1 Affine Decryption

Decryption Matrix

\[ D \cdot \begin{pmatrix} V^* \\ V_{tag} \end{pmatrix} = \begin{pmatrix} C \\ V \end{pmatrix} = \begin{pmatrix} U^* = \begin{pmatrix} U \\ U_{tag} \end{pmatrix} \\ Z = \begin{pmatrix} M \\ T \end{pmatrix} \end{pmatrix} \]

- \( D \): Dec Matrix
INT-RUP Attack on Affine Mode AE

Queries of INT-RUP Adversary

- **Encryption Query**: i/p: \((N, AD, M^0 = (M^0_1, \ldots, M^0_l))\) 
o/p: \(C^0 = (C^0_1, \ldots, C^0_l, T^0)\)

- **Unverified Plaintext Query**: i/p: \((N, AD, C^1 = (C^1_1, \ldots, C^1_l))\). 
o/p: \(M^1 = (M^1_1, \ldots, M^1_l)\)

- **Forged Query**: \((N, AD, C^f = (C^f_1, \ldots, C^f_l), T^f)\), which realizes a \(\delta = (\delta_1, \ldots, \delta_l)\) sequence.

Question

How to Compute \(C^f\) and \(T^f\) ???

Mridul Nandi
Authenticated Encryption
Different Terms

- \( U^0 = (U_0^0, \ldots, U_i^0) \), \( V^0 = (V_0^0, \ldots, V_i^0) \)
- \( U^1 = (U_0^1, \ldots, U_i^1) \), \( V^1 = (V_0^1, \ldots, V_i^1) \)
- \( U^f = (U_0^f, \ldots, U_i^f) \), \( V^f = (V_0^f, \ldots, V_i^f) \)
- \( \Delta M^{01} = M_0 + M_1, \Delta C^{01} = C_0 + C_1 \)
- \( \Delta M^{0f} = M_0 + M_f, \Delta C^{0f} = C_0 + C_f \)
Compute $C^f$ and $T^f$

- $C^f$ realizes a $\delta = (\delta_1, \ldots, \delta_l)$-sequence. $\forall i \leq l$, $U_i^f = U_i^{\delta_i}$ and $\forall i > l$, $U_i^f = U_i^0$.

- $\delta = (\delta_1, \delta_2, \delta_3, \delta_4, \delta_5) = (0, 1, 0, 0, 1)$
- $U^f = (U_1^0, U_2^1, U_3^0, U_4^0, U_5^1)$ and $V^f = (V_1^0, V_2^1, V_3^0, V_4^0, V_5^1)$
- $\delta$ helps to compute $C^f$, $T^f$ (Described later)
Solve for $\delta$

Can compute $\Delta U^{01}$ and $\Delta V^{01}$ from $\Delta C^{01}$ and $\Delta M^{01}$
Solve for $\delta$

$\delta$ makes $\Delta U^0f = 0$ (Collision between $U^0f$ and $U^f$)

Computed by $\delta$, $\Delta U^01$, $\Delta V^01$

- Only way - Collision between $U^0$ (We know $T^0$) and $U^f$
- Solve $\delta$ such that collision between $U^0$ and $U^f$
- $\Delta U^0f = \delta \cdot \Delta U^01$, $\Delta V^0f = \delta \cdot \Delta V^0f$
How to Solve $\delta$

From Decryption

- $\Delta C^0_f = Lin_1(\Delta U^0_f, \Delta V^0_f) = \delta . Lin_1(\Delta U^{01}, \Delta V^{01})$

- Solve $\Delta U^0_f = Lin_2(\Delta C^0_f, \Delta V^0_f) = \delta . Lin'_2(\Delta U^{01}, \Delta V^{01}) = 0$

- Solution - $\delta^*$
Compute $\Delta C^{0f}$

$$\Delta C^{0f} = \delta^*.Lin_1(\Delta U^{01}, \Delta V^{01})$$

Compute $C^f$

$$C^f = C^0 + \Delta C^{0f}$$

Compute $\Delta T^f$

$$\Delta T^{0f} = Lin_3(\Delta C^{0f} + \Delta V^{0f}) \ (\Delta C^0, \Delta V^{0f} \text{ known}, \Delta V_{tag}^{0f} = 0)$$

Compute $T^f$

$$T^f = T^0 + \Delta T^{0f}$$
A generic Attack on Rate-1 Affine AE

Significance of the Result

- A Geneic result showing INT-RUP insecurity of “Rate-1” Affine mode AE.
- Guideline: To achieve INT-RUP security, one has to compromise efficiency.

Interesting Question?

- How much Efficiency should we degrade?
- Lets Analyze some “High rate” (< 1) popular AE construction.
Revisit CPFB (Rate 3/4)
INT-RUP Attack on CPFB

Encryption query: i/p: \((N, A, M^0)\), \(|M^0| = l = 129\). o/p: \(C^0\)

Unverified Plaintext decryption query: i/p: \((N, A, C^1)\) of length \(l\). o/p: \(M^1\)

Compute \(Y\) values: \(Y_1^0, \ldots, Y_i^0\) and \(Y_1^1, \ldots, Y_i^1\) from the two queries (by \(M^0 + C^0\) and \(M^1 + C^1\)).

Find the \(\delta\)-sequence: \(\delta = (\delta_1, \ldots, \delta_l)\), with \(\delta_1 = 0\) such that,

\[
\sum_{i=2}^l (Y_i^{\delta_i} \parallel Z_i^{\delta_i}) = \sum_{i=2}^l (Y_i^0 \parallel Z_i^0).
\]

Expect \(2^{32}\)-many such \(\delta\)-sequences.
INT-RUP Attack on CPFB

Perform the following for all such \( \delta \)-sequence

Set \( C^f_1 = C^0_1 \). For all \( 1 < i < l \), set \( C^f_i = C^\delta_i \) if \( \delta_{i-1} = \delta_i \) and \( C^\delta_i + Y^0_i + Y^1_i \), otherwise.

Set \( C^f_l = C^0_l \) if \( \delta_l = 0 \). Else, set \( C^f_l = C^0_l + Y^0_l + Y^1_l \).

Return \((C^f_1, C^f_2, \cdots, C^f_l, T^0)\) as forged Ciphertext.
How to resist the INT-RUP Attack?

Potential Weakness of CPFB

1. $Y_i$ values can be observed. Only $Z_i$-values are unknown.
2. $Z_i$ has only 32-bit entropy on the Tag.

Removing the Weakness

- Ensure 128-bit entropy of $Z$-values on the tag.
- Ensure at-least 4 different $Z$-values for 2 messages of same length.
mCPFB: modified CPFB

Introduce ECC Code

Expand $M = (M_1, \ldots, M_l)$ by a Distance 4 Error Correcting Code

$\text{ECCode} :$

$\text{ECCode}(M) = (M_1, \ldots, M_l, M_{l+1}, M_{l+2}, M_{l+3})$

$(M_{l+1}, M_{l+2}, M_{l+3}) = V^{(3,l)}_\beta \cdot M$

Produce 128-bit entropy of $Z$-values during Tag Generation:

Update $Z^M$ as follows:

$Z_M = V^{(4,l+3)}_\alpha \cdot (Z_2, Z_3, \ldots, Z_{l+3}, Z_{l+4}) \oplus (0^{32}|| (Y_2 \oplus \cdots \oplus Y_{l+3}))$
mCPFB: modified CPFB

Changes in the keys

- $\kappa_0$ is used as the masking key only.
- $\kappa_1$ is used as block-cipher key for AD processing.
- $\kappa_1, \ldots, \kappa_{-2}$ is used as block-cipher keys for message processing.
- $\kappa_{-1}$ is used as block-cipher key for tag and producing $L$-values.
INT-RUP Security of mCPFB

Claim 1

Consider the function $f$ that takes $N$, $I$ and $i$ as input and outputs $O$ such that $O = E_{\kappa[i]}(I || (i \mod 2^{32}) + \kappa_0)$ where $\kappa[i] = E_K(N || j || I)$, $j = \lceil \frac{i}{2^{32}} \rceil$. $f$ is assumed to have $(q, \epsilon)$-PRF security where $\epsilon$ is believed to achieve beyond birthday security.

INT-RUP advantage

$f$: $(q_e + q_r, \epsilon)$-PRF. Any adversary $A$ with $q_e$ many encryption query and $q_r$ many unverified plaintext queries, one forgery attempts, has the advantage:

$$\text{Adv}^{\text{int-rup}}_{m\text{CPFB}}(A) \leq \frac{5}{2^{128}} + \epsilon$$
Proof Sketch

Argument for Different Cases

- **(Case A)** \( \forall i, N^* \neq N_i \): Through randomness of \( \kappa_{-1} \).
- **(Case B)** \( \exists \text{ unique } i \ni N^* = N_i, T^* \neq T_i \): Through randomness of \( \kappa_{-1} \).
- **(Case C)** \( \exists \text{ unique } i \ni N^* = N_i, T^* = T_i, |C_i| = |C^*| \): Through randomness of \( Z_i \)’s.
- **(Case D)** \( \exists \text{ unique } i \ni N^* = N_i, T^* = T_i, |C_i| \neq |C^*| \): Through randomness of \( \kappa_{-1} \).
INT-RUP Analysis of “High rate” Affine Mode AE

Significance of The Result

- Any “rate-1” affine mode AE is INT-RUP Insecure.
- INT-RUP comes with small degrade in efficiency.
- “Rate-1” is a borderline criteria for INT-RUP security.
1. Introduction to Authenticated Encryption.
2. Security Notion of Authenticated Encryption
4. INT-RUP Analysis for “High-rate” Affine Constructions
5. Another Fault Based Almost Universal Forgery on CLOC-SILC
   - Single Bit Fault Based Forgery on CLOC
   - Single Bit Fault Based Forgery on SILC
   - Implementation of Fault
Description of CLOC

\[ V \leftarrow \text{Hash}_K(N, M), \quad C \leftarrow \text{Enc}_K(V, M), \quad T \leftarrow \text{PRF}_K(V, C) \]
Description of SILC

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

$$V \leftarrow Hash_K(N, M), C \leftarrow Enc_K(V, M), T \leftarrow PRF_K(V, C)$$
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 $(X, Y), (X_1, Y_1), (X_2, Y_2)$
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 \(a - 1\) enc queries
- Find \(A'\) colliding with \(A\) at \(V\)
- Enc query: \((N, A', M) \rightarrow (C, T)\)
- Forge with \((N, A, C, T)\)
1. Introduction to Authenticated Encryption.

2. Security Notion of Authenticated Encryption


4. INT-RUP Analysis for “High-rate” Affine Constructions

5. Another Fault Based Almost Universal Forgery on CLOC-SILC
   - Single Bit Fault Based Forgery on CLOC
   - Single Bit Fault Based Forgery on SILC
   - Implementation of Fault
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 \(a\) enc queries
- Find \(A'\) colliding with \(A\) at \(V\)
- Enc query: \((N, A', M) \rightarrow (C, T)\)
- Forge with \((N, A, C, T)\)
Fault Attack Setup

PC

FGPA Board

Oscilloscope

Electromagnetic Probe

Delay Generator

RF Amplifier

RF Generator

Transmission

Reception

Trigger

Electromagnetic Pulse Injection

Amplified Electromagnetic Pulse

Trigger

Trigger

Trigger

Electromagnetic Pulse
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
Thank you