Security Analysis of HyENA Authenticated Encryption Mode

Avik Chakraborti[1], Nilanjan Datta[2], Ashwin Jha[2], Snehal Mitragotri[2], Mridul Nandi[2]

[1] NTT Secure Platform Laboratories, Japan
[2] Indian Statistical Institute, Kolkata, India
chakraborti.avik@lab.ntt.co.jp,nilanjan_isi_jrf@yahoo.com,ashwin.jha1991@gmail.com,snehal.mitr@gmail.com,mridul.nandi@gmail.com

Abstract. In CHES 2017, Chakraborti et al. proposed COFB, a combined feedback and block cipher based rate 1 authenticated encryption (AE) with only $3n/2$-bit state, which is optimal for any secure rate 1 block cipher based AE. Later, another optimal state authenticated encryption scheme denoted HyENA has been submitted to NIST Lightweight Cryptography standardization project. HyENA is even better than COFB in terms of the XOR-count. However, the specification clearly provides security claims in terms of privacy and integrity. This write up specifies a nominally updated version of HyENA and justifies the same security levels claimed by the designers. We provide a detailed AE security analysis. The updates are done to optimize the hardware area further in the multiplexer level. The security level of HyENA is exactly the same as COFB. The proof approach for HyENA is based on Patarin's H Coefficient technique similar to COFB but it needs a different handling of the case analysis. In addition to the theoretical security analysis, we also provide a rough benchmark with the existing designs in terms of the gate counts for the XORs.

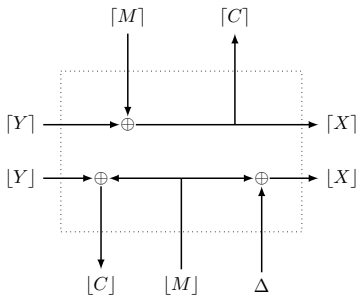
Keywords: feedback based AE · HyENA · lightweight · hybrid feedback

1 HyENA Authenticated Encryption Mode

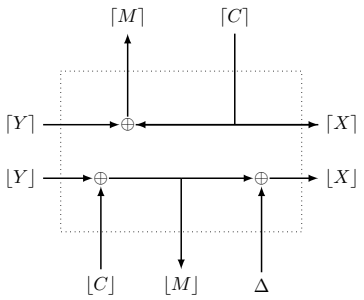
In this section, we present a complete specification of a nominally updated version of the HyENA mode [5]. We have slightly updated the Δ update function (mainly how the power of α and $1+\alpha$ is computed) to have a uniformity in the Δ update function for both the associated data and the message processing phase. This consequently optimizes the hardware circuit complexity further. We also give detailed algorithmic descriptions for this mode. The changes are made in line 3, 13 of the Proc-AD algorithm and line 10 of the Proc-TXT algorithm in Fig 4. HyENA encryption mode receives an encryption key $K \in \{0,1\}^\kappa$, a nonce $N \in \{0,1\}^r$, an associated data $A \in \{0,1\}^*$, and a message $M \in \{0,1\}^*$ as inputs, and returns a ciphertext $C \in \{0,1\}^{|M|}$ and a tag $T \in \{0,1\}^n$. The decryption algorithm receives a key $K \in \{0,1\}^\kappa$, an associated data $A \in \{0,1\}^*$, a nonce $N \in \{0,1\}^r$, a ciphertext $C \in \{0,1\}^*$ and a tag $T \in \{0,1\}^n$ as inputs and returns the plaintext $M \in \{0,1\}^{|C|}$, corresponding to the ciphertext C, if the tag T authenticates.

1.1 Hybrid FeedBack Function

Here we describe the hybrid feedback function. The pictorial description of hybrid feedback function can also be found in Figure 3. The feedback function is illustrated in Figure 1 and 2.

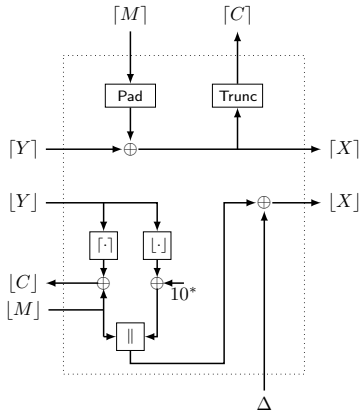


(a) HyFB+ module.

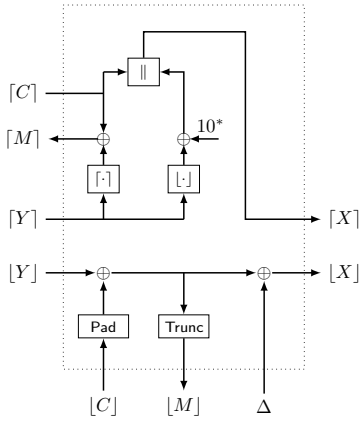


(b) HyFB- module.

Figure 1: HyFB module of HyENA for full data blocks. The number of XOR count is equals to $3n/2$.



(a) HyFB+ module.



(b) HyFB- module.

Figure 2: HyFB module of HyENA for partial data blocks. Similar to the full block case, the number of XOR count here also equals to $3n/2$.

1.2 HyENA Mode

Here we provide the specification of HyENA mode. Complete specification of HyENA is presented in Algorithm 4.

1.2.1 Initialization

We define the initial state as

$$IV \leftarrow N \| 0^{n-r-2} \| b_0 \| b_1,$$

where b_0 is a bit indicating whether the associated data is empty or not ($b_0 = 0$ iff associated data is empty) and b_1 is a bit indicating whether there is any data or not

($b_1 = 0$ iff both associated data and plaintext are empty). Note that, the combination $(b_0, b_1) = (1, 0)$ is impossible. This initial vector is encrypted to generate the initial state $Y[0]$. The least significant $n/2$ bits of $Y[0]$ is considered as the masking value Δ. Formally, we define

$$\Delta \leftarrow \lfloor E_K(IV) \rfloor.$$

1.2.2 Associated Data Processing

For associated data processing, first we parse the associated data in n-bit blocks. We perform an 10^* padding on the associated data in the following cases: (i) when associated data is empty, and (ii) when final associated data block is partial. After the padding is done, we process the associated data blocks sequentially and updates the state and the masking value as follows:

$$\begin{aligned} \Delta &\leftarrow 2 \cdot \Delta\,, \\ (X[i], \star) &\leftarrow \textsc{HyFB} + (Y[i-1], A[i], \Delta[i])\,, \\ Y[i] &\leftarrow E_K(X[i])\,, \end{aligned}$$

where $\star$ denotes some value that we do not bother. To process the final associated data block, we multiply Δ by 3 (for full) or 3^2 (for partial) for the purpose of domain separation.

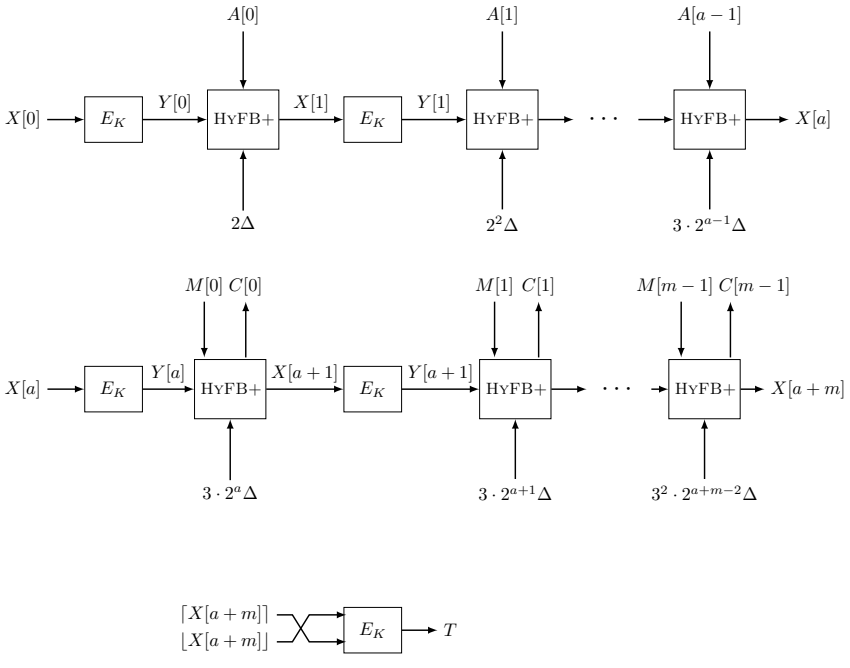


Figure 3: HyENA authenticated encryption mode for full data blocks.

1.2.3 Plaintext Processing

For plain text processing, first we parse the plain text in n-bit blocks. We perform an 10^* padding on the plain text only when the final plain text block is partial. Note that, we do not perform any operation if plaintext is empty. After the padding is done, we process the plaintext blocks sequentially and updates the state and the masking value as follows:

$$\begin{aligned} \Delta &\leftarrow 2 \cdot \Delta\,, \\ (X[a+i], C[i]) &\leftarrow \textsc{HyFB} + (Y[a+i-1], M[i], \Delta[i])\,, \\ Y[a+i] &\leftarrow E_K(X[a+i]). \end{aligned}$$

To process the final plain text block, we multiply Δ by 3 (for full) or 3^2 (for partial) for the purpose of domain separation.

1.2.4 Tag Generation

To generate the tag, we swap the most significant and least significant $n/2$ bits of the state and performs a block cipher encryption:

$$T \leftarrow E_K(\lfloor X[a+m] \rfloor \| \lceil X[a+m] \rceil)$$

1.3 Recommended Instantiation

In this section, we recommend an instance of HyENA. HyENA is parametrized by the choice of the underlying block cipher E and the size of the nonce in bits r. We instantiate the block cipher with GIFT-128/128 [3] with an 128-bit key and an 128-bit block. Note that, the block cipher is well defined and in this paper we do not include any description of GIFT-128/128. r is chosen to be 96. The details of the security analysis of GIFT-128/128 are detailed in [4, 3].

2 Comparison with COFB

In this section, we provide a rough comparison in gate counts between HyENA amnd COFB. We follow the calculaton from [2] to get the approximate gate count for COFB. We adopt the same assumption to count the hardware gate counts for HyENA. Note that, we estimate encryption-only circuit.

COFB aims to reduce the hardware area for a rate 1 construction by optimizing the state size. HyENA too optimizes the state size and maintains exactly the same storage size. It maintains an n-bit state to hold the block for the underlying block cipher and uses an additional $n/2$-bit state to hold a secret nonce-dependent mask. However, JAMBU also uses an 1.5-bit state but with rate 1/2. SUNDAE optimizes the state size to n-bit for a rate 1/2 construction. We count the gates by dividing the hardware circuit of HyENA into two parts.

- **Register Gate Count:** When instantiated with an 128-bit block cipher, we need a 64-bit register for the secret mask. At each round, it is multipied with α, $(1+\alpha)$ or $(1+\alpha)^2$. However, multiplication by $(1+\alpha)^2$ can be implemented with multiplication with $(1+\alpha)$ in two successive clock cycles. The underlying primitive polynomial is chosen as $x^{64}+x^4+x^3+x+1$ such that only 3 XORs are required for α-multiplication and $64+3=67$ XORs are needed for $1+\alpha$-multiplication. We can use scan register to efficiently implement the circuit. The overall estimate of the gate count is 612 gate counts (350 for 64-bit scan register with 5.5 gates for each bit, 128 for 64-bit multiplexer and 134 for XOR gates). Note that, this is exactly the same as COFB.

Algorithm HyENA-Enc(K, N, A, M)

1. $Y \leftarrow \textsc{Init}(N, A, M)$
2. $(X, \Delta) \leftarrow \textsc{Proc-AD}(Y, A)$
3. **if** $|M| \neq 0$ **then**
4. $\quad (X, C) \leftarrow \textsc{Proc-TXT}(X, \Delta, M, +)$
5. $T \leftarrow \textsc{Tag-Gen}(X)$
6. **return** (C, T)

Algorithm $\textsc{Init}(N, A, M)$

1. $b_0 \leftarrow (|A| = 0)?\ 1 : 0$
2. $b_1 \leftarrow (|A| + |M| = 0)?\ 1 : 0$
3. $Y \leftarrow E_K(N \| 0^{n-r-2} \| b_1 \| b_0)$
4. **return** Y

Algorithm $\textsc{Proc-AD}(Y, A)$

1. $\Delta \leftarrow \lfloor Y \rfloor$
2. **if** $|A| = 0$ **then**
3. $\quad \Delta \leftarrow 3^2 \odot \Delta$
4. $\quad (X, \star) \leftarrow \textsc{HyFB+}(Y, \Delta, 0^{n-1}1)$
5. $\quad$ **return** (X, Δ)
6. **else**
7. $\quad (A_{a-1}, \ldots, A_0) \xleftarrow{n} A$
8. $\quad$ **for** $i = 0$ **to** $a - 2$
9. $\quad\quad \Delta \leftarrow 2 \odot \Delta$
10. $\quad\quad (X, \star) \leftarrow \textsc{HyFB+}(Y, \Delta, A_i)$
11. $\quad\quad Y \leftarrow E_K(X)$
12. $\quad t \leftarrow (|A_{a-1}| = n)?\ 1 : 2$
13. $\quad \Delta \leftarrow 3^t \odot \Delta$
14. $\quad (X, \star) \leftarrow \textsc{HyFB+}(Y, \Delta, A_{a-1})$
15. $\quad$ **return** (X, Δ)

Algorithm Tag-Gen(X)

1. $T \leftarrow E_K(\lfloor X \rfloor \| \lceil X \rceil)$
2. **return** T

Algorithm HyENA-Dec(K, N, A, C, T)

1. $Y \leftarrow \textsc{Init}(N, A, M)$
2. $(X, \Delta) \leftarrow \textsc{Proc-AD}(Y, A)$
3. **if** $|C| \neq 0$ **then**
4. $\quad (X, M) \leftarrow \textsc{Proc-TXT}(X, \Delta, C, -)$
5. $T' \leftarrow \textsc{Tag-Gen}(X)$
6. **if** $T' = T$ **then return** M
7. **else return** $\perp$

Algorithm $\textsc{HyFB+}(Y, \Delta, M)$

1. $C \leftarrow \mathsf{Trunc}_{|M|}(Y) \oplus M$
2. $\overline{M} \leftarrow \mathsf{Pad}(M)$, $\overline{C} \leftarrow \mathsf{Pad}(C)$
3. $B \leftarrow \lceil \overline{M} \rceil \| (\lfloor \overline{C} \rfloor \oplus \Delta)$
4. $X \leftarrow B \oplus Y$
5. **return** (X, C)

Algorithm $\textsc{HyFB-}(Y, \Delta, C)$

1. $M \leftarrow \mathsf{Trunc}_{|C|}(Y) \oplus C$
2. $\overline{M} \leftarrow \mathsf{Pad}(M)$, $\overline{C} \leftarrow \mathsf{Pad}(C)$
3. $B \leftarrow \left(\lceil \overline{M} \rceil \| (\lfloor \overline{C} \rfloor \oplus \Delta) \right)$
4. $X \leftarrow B \oplus Y$
5. **return** (X, M)

Algorithm $\textsc{Proc-TXT}(X, \Delta, D, \mathsf{dir})$

1. $(D_{d-1}, \ldots, D_0) \xleftarrow{n} D$
2. **for** $i = 0$ **to** $d - 2$
3. $\quad \Delta \leftarrow 2 \odot \Delta$
4. $\quad Y \leftarrow E_K(X)$
5. $\quad$ **if** $\mathsf{dir} = +$ **then**
6. $\quad\quad (X, O_i) \leftarrow \textsc{HyFB+}(Y, \Delta, D_i)$
7. $\quad$ **else**
8. $\quad\quad (X, O_i) \leftarrow \textsc{HyFB-}(Y, \Delta, D_i)$
9. $t \leftarrow (|D_{d-1}| = n)?\ 1 : 2$
10. $\Delta \leftarrow 3^t \odot \Delta$
11. $Y \leftarrow E_K(X)$
12. **if** $\mathsf{dir} = +$ **then**
13. $\quad (X, O_{d-1}) \leftarrow \textsc{HyFB+}(Y, \Delta, D_{d-1})$
14. **else**
15. $\quad (X, O_{d-1}) \leftarrow \textsc{HyFB-}(Y, \Delta, D_{d-1})$
16. **return** $(X, (O_{d-1} \| \ldots \| O_0))$

Figure 4: Formal Specification of HyENA Authenticated Encryption and Decryption algorithm. We use the notation $\star$ to denote values that we do not care.

- **Feedback Function Gate Count:** COFB uses a G function which divides the 128-bit AES state into two 64-bit parts, swaps them and left rotate the right part.

This function does not need any additional gates and can be implemented with wire shuffling. However, the linear feedback matrix for HyENA also does not need XOR to compute.

Consider Y to be the AES block cipher output state and G be the feedback matrix such that next feedback is computed as $G \cdot Y \oplus M$ (M is the message block). Let Y is parsed as $(Y[1], Y[2])$ where $|Y[1]| = |Y[2]| = 64$. For COFB, the feedback is computed as $Y[2] \| (Y[1] <<< 1) \oplus M \oplus \Delta \| 0^{64}$ and the ciphertext block C is computed as $C = Y \oplus M$. let M is parsed as $(M[1], M[2])$. For HyENA, the feedback is computed as $(Y[1] \oplus M[1]) \| M[2] \oplus 0^{64} \| \Delta$ and the ciphertext block C is computed as $C = Y \oplus M$. Thus, overall, gate count for the feedback computation in case of COFB is 640 (total 128 + 128 + 64 = 320 XORs). For HyENA, the gate count is 512 (total 64 + 128 + 64 = 256 XORs).

Overall, the total gate count for COFB is GIFT Core + Register Gate Count + 640 + 250 (control logic) + *Length counter* and for HyENA is GIFT Core + (350 + 128 + 134) + 512 + 250 (control logic) + *Length counter*.

3 Security

In this section, we provide the security of HyENA, mainly we prove the following Theorem:

Theorem 1.

$$\begin{aligned}\mathbf{Adv}^{\mathrm{AE}}_{\mathrm{HyENA}}(q_e, q_v, \sigma_e, \sigma_v, t) \leq & \mathbf{Adv}^{\mathrm{prp}}_{E_K}(q', t') + \frac{2\sigma_e}{2^{n/2}} + \frac{\sigma_e^2}{2^n} + \frac{max\{n, nq_e/2^{n/4}\}}{2^{n/4}} \\ & + \frac{nq_e}{2^{n/2}} + \frac{max\{n, nq_e/2^{n/4}\}q_e}{2^{3n/4}} + \frac{3nq_v}{2^{n/2}} \\ & + \frac{2max\{n, nq_e/2^{n/4}\}q_v}{2^{3n/4}} + \frac{nq_v}{2^{3n/4}} + \frac{q_v}{2^n} + \frac{2n\sigma_v}{2^{n/2}}.\end{aligned}$$

where $q' = q_e + \sigma_e + q_v + \sigma_v$ which corresponds to the total number of block cipher calls through the game and $t' = t + O(q')$.

Proof. Without loss of generality, we can assume that $q' \leq 2^{n/2-1}$, since otherwise the right hand side becomes more than 1. The first transition we make is to use an n-bit (uniform) random permutation P instead of E_k and then use an n-bit (uniform) random function R instead of P. The probabilities corresponding to these two transitions are given in the first two terms of our bound. They are derived from the standard PRP-PRF switching lemma and from the computation of information theoretic reduction. The rest of the proof is done in the following sections.

3.1 Notations and Set-up

Fix a deterministic non-repeating query making distinguisher adv that interacts with either (1) the real oracle or (2) the ideal oracle making at most

1. q_e encryption queries $(N_i^+, A_i^+, M_i^+)_{i=1..q_e}$ with an aggregate of total σ_e many blocks and
2. attempts to forge with q_v many queries $(N_i^-, A_i^-, C_i^-, T_i^-)_{i=1..q_v}$ having a total of σ_v many blocks.

We assume that $\forall i$, M_i^+ and A_i^+ have m_i^+ and a_i^+ blocks respectively and C_i^- and A_i^- have c_i^- and a_i^- blocks respectively. We use the notation X, Y to denote the intermediate variables. Let

$$(S_i[0], S_i[1], \cdots, S_i[l_i^+ - 1]) \leftarrow (A_i^+[0], \cdots, A_i^+[a_i^+ - 1], M_i^+[0], \cdots, M_i^+[m_i^+ - 1])$$

where $l_i^+ = a_i^+ + m_i^+$. If bitwise representation of n-bit string G is $(G_{n-1} \cdots G_0)$. Then for $u > w$, we denote $(G_u \cdots G_w)$ by G_{u-w} which is a $u - w + 1$-bit substring of G from u^{th} bit to w^{th} bit of G.

3.2 Overview of the Attack Transcript

We begin with a description of the ideal oracle which consists of two phases.

- **Online phase**: For the i^{th} encryption query $(N_i^+, A_i^+ = (A_i[0], \ldots, A_i^+[a_i-1]), M_i^+ = (M_i^+[0], \ldots, M_i^+[m_i-1]))$, the oracle samples $(Y_i^+[a_i^+], \ldots, Y_i^+[l_i^+]) \leftarrow_\$ \{0,1\}^{n(m_i^+ + 1)}$ independently. It next sets the tag $T_i^+ = Y_i^+[l_i^+]$ and $C_i^+ = (C_i^+[0], \ldots, C_i^+[m_i^+ - 1])$ where $C_i^+[j] = Y_i^+[j + a_i^+] \oplus M_i^+[j]$ for $0 \le j \le m_i^+ - 1$ and returns (C_i^+, T_i^+) to $\mathcal{A}$.
- **Offline phase**: After $\mathcal{A}$ makes all the queries the oracle samples other Y^+ values as $Y_i^+[j] \leftarrow_\$ \{0,1\}^n$, for $0 \le j \le a_i - 1$.

For convenience, we slightly modify the experiment where we reveal to the adversary $\mathcal{A}$ (after $\mathcal{A}$ made all its queries and obtains corresponding responses but before it outputs its decision) the Y^+-values and now the adversary can set all intermediate values $X_i^+[j]$ using $S_i[j]$ and $Y_i^+[j]$. Note that $\Delta_i^+ = \lfloor Y_i^+[0] \rfloor$ and $\Delta_i^- = \lfloor Y_i^-[0] \rfloor$.
Overall, the transcript of the adversary $\tau := (\tau_e, \tau_v)$ be the list of queries and responses of $\mathcal{A}$ that constitutes the query response transcript of $\mathcal{A}$, where

- $\tau_e = (N_i^+, A_i^+, M_i^+, X_i^+, Y_i^+, T_i^+)_{i=1..q_e}$,
- $\tau_v = (N_j^-, A_j^-, C_j^-, T_j^-, \perp)_{j=1..q_v}$.

A prefix for a decryption query is defined as the common prefix blocks between the decryption query input string and an encryption query (if any) output string prepended with the nonce and the associated data. The length of the longest common prefix for the i^{th} decryption query is denoted as p_i. Note that if the decryption query uses a fresh nonce (not occurred during encryption queries), then it does not share any common prefix with any of the encryption queries then we set $p_i = -1$.

By ip_{ideal} and ip_{real} we denote the interpolation probability distribution of transcript τ induced by the ideal world and real world respectively. Note that, we use Patarin's Coefficient H technique and below we start the proof by first identifying the bad events.

3.3 Identifying and Bounding Bad Events

Now, we define a set of events (initial bad events) for which the adversary aborts.

(i) B1: $\mathsf{mColl}(\Lambda) > n$ where Λ is the tuple of all $\lfloor X_i^+[l_i^+] \rfloor$ and $\lceil X_i^+[j] \rceil$ values for $0 \le j < l_i$.
This event signifies that n-multi-collision occurs in the upper part of the inputs of all blocks except the last blocks and the lower part of the inputs of all last blocks corresponding to the encryption queries.

(ii) B2: $\mathsf{mColl}(X^+[l]_{(n/2-1)-n/4}) > c$ where $c = \lceil nq_e/2^{n/4} \rceil$ and $X^+[l]_{(n/2-1)-n/4}$ is the tuple of all $X_i^+[l_i^+]_{(n/2-1)-n/4}$ values (it is the second $n/4$-bit chunk of $X_i^+[l_i^+]$. For example, when $n = 128$ it is $X_i^+[l_i^+]_{63-32}$) for $i \in [1 \cdots q_e]$,

(iii) B3: $X_i^+[j] = X_{i'}^+[0]$ for some $i, i' \in [1 \cdots q_e]$ and $0 < j < l_i^+$.

(iv) B4: $X_i^+[l_i^+] = X_{i'}^+[0]$ for some $i, i' \in [1 \cdots q_e]$

(v) B5: $X_i^+[j] = X_{i'}^+[j']$ for some $(i, j) \ne (i', j')$ and $j, j' > 0$.

(vi) B6: $X_i^-[p_i+1] = X_{i'}^+[j']$ for some i, i' and $0 < j' < l_{i'}^+$.

(vii) B7: $X_i^-[p_i+1] = X_{i'}^+[l_{i'}^+]$ for some i and i'.

(viii) B8: $X_i^-[p_i+1] = X_{i_1}^+[0]$ and $X_i^-[p_i+2] = X_{i'}^+[j']$ for some $p_i(\geq 0), i, i', i_1$ and $0 < j' \leq l_{i'}^+$.

The following lemma bounds the probability of bad transcripts in ideal oracle:

Lemma 1. *For any transcript τ,*

$$\begin{aligned} ip_{ideal}(\tau \in \mathcal{V}_{bad}) &\leq Pr[B1] + Pr[B2] + Pr[B3 \wedge B1^c] + Pr[B4 \wedge B2^c] + Pr[B5] \\ &\quad + Pr[B6 \wedge B1^c] + Pr[B7 \wedge B1^c \wedge B2^c] + Pr[B8 \wedge B1^c] \\ &\leq \frac{2\sigma_e}{2^{n/2}} + \frac{\sigma_e^2}{2^n} + \frac{max\{n, nq_e/2^{n/4}\}}{2^{n/4}} + \frac{nq_e}{2^{n/2}} + \frac{max\{n, nq_e/2^{n/4}\}q_e}{2^{3n/4}} + \frac{3nq_v}{2^{n/2}} \\ &\quad + \frac{2max\{n, nq_e/2^{n/4}\}q_v}{2^{3n/4}} + \frac{nq_v}{2^{3n/4}} \end{aligned}$$

Proof. Throughout the proof, we assume that all probabilities are defined over the ideal game. Here we provide the upper bounds for the bad events (in ideal oracle) one by one, as follows:

1. **Pr[B1].** The event B1 is a multi-collision event for randomly chosen n many $n/2$-bit strings out of σ_e many $n/2$-bit strings. As the Y^+-values are sampled uniformly and independently in the ideal game, we have,

$$Pr[B1] \leq \frac{\binom{\sigma_e}{n}}{2^{n/2(n-1)}} \leq \left(\frac{2\sigma_e}{2^{n/2}}\right)^n \leq \frac{2\sigma_e}{2^{n/2}}$$

   The last inequality follows from the assumption that $\sigma_e \leq 2^{n/2-1}$

2. **Pr[B2]**.This event is a multicollision event for randomly chosen c ($\lceil nq_e/2^{n/4} \rceil$) many $n/4$-bit strings out of q_e many $n/4$-bit strings. Hence,

$$Pr[B2] \leq 2^{n/4}\left(\frac{\mathrm{e}q_e}{c2^{n/4}}\right)^c \leq 2^{n/4}\left(\frac{\mathrm{e}}{n}\right)^{\max\{n, nq_e/2^{n/4}\}} \leq 2^{n/4}\left(\frac{1}{2^n}\right) \leq \frac{1}{2^{3n/4}}.$$

   The first inequality follows from the well-known results on multicollision [7, 1] and e is the Euler's number. The third inequality follows from the assumption that $n \geq 2\mathrm{e}$.

3. **Pr[B3 $\wedge$ B1^c]**. Fix a pair of integers (i, j) such that $i \in [1 \cdots q_e]$. Then for $j < l_i^+$, we have

$$X_i^+[j] = N_{i'}^+ || 0^{n-r-2} || b_{1i'}^+ b_{0i'}^+$$

   where r is the nonce size and we can express $\lfloor X_i^+[j] \rfloor$ as $\lfloor S_i[j] \rfloor \oplus 2^a \odot \Delta_i^+$ for some constant a. Here, Δ_i^+ is uniformly distributed. Note that there could be at most n many (i, j) indices for each i' and there are q_e many i' indices. Hence,

$$Pr[B3 \wedge B1^c] \leq \frac{nq_e}{2^{n/2}}.$$

4. **Pr[B4$\wedge$ B2^c]**. Fix an integer i such that $i \in [1 \cdots q_e]$. Then for $j = l_i^+$ and for some constants a, b, we have

$$\lceil N_{i'}^+ \rceil_{n/2} = \lfloor S_i[l_i^+ - 1] \rfloor \oplus 2^a \odot 3^b \odot \Delta_i^+ (= \lceil X_i^+[l_i] \rceil)$$

$$\lfloor N^+_{i'} \rfloor_{n/4} || 0^{n-r-2} || b^+_{1i'} b^+_{0i'} = \lceil S_i[l^+_i - 1] \rceil \oplus \lceil Y^+_i[l^+_i - 1] \rceil (= \lfloor X^+_i[l_i] \rfloor)$$

Here, Δ^+_i and $\lceil Y^+_i[l^+_i - 1] \rceil$ are independent and uniformly distributed. Also i' can take at most q_e many values and B2^c implies that i can take at most c many values. Hence,

$$Pr[B4 \wedge B2^c] \leq \frac{cq_e}{2^{3n/4}} \leq \frac{max\{n, nq_e/2^{n/4}\}q_e}{2^{3n/4}}$$

5. **Pr[B5].** For any $(i,j) \neq (i',j')$ and $j, j' > 0$, we have the following three possibilities: Case(i): $j < l^+_i, j' < l^+_{i'}$ Then for any $(i,j) \neq (i',j')$,the event $X^+_i[j] = X^+_{i'}[j']$ is nothing but two non-trivial linear equations. One is on $\lceil Y^+_i[j-1] \rceil$ & $\lceil Y^+_{i'}[j'-1] \rceil$ and other is on $\text{Const}_j \odot \Delta^+_i$ & $\text{Const}_{j'} \odot \Delta^+_{i'}$ for some constants Const_j & $\text{Const}_{j'}$. For $i \neq i'$, we have $\lceil Y^+_i[j-1] \rceil, \lceil Y^+_{i'}[j'-1] \rceil, \Delta^+_i$ and $\Delta^+_{i'}$ are independent and uniformly distributed. For $i = i'$, we have $\text{Const}_j \neq \text{Const}_{j'}$ and $\lceil Y^+_i[j-1] \rceil, \lceil Y^+_{i'}[j'-1] \rceil$ are independent and uniformly distributed. Hence this event has probability at most 2^{-n}. Therefore,

$$Pr[X^+_i[j] = X^+_{i'}[j']] \leq \frac{(\sigma_e - q_e)^2}{2^n}.$$

Similarly, we can argue for the other cases also.
Case(ii): For $j < l^+_i, j' = l^+_{i'}$ we have

$$Pr[X^+_i[j] = X^+_{i'}[j']] \leq \frac{(\sigma_e - q_e)q_e}{2^n}$$

Case(iii): For $j = l^+_i, j' = l^+_{i'}$, we have

$$Pr[X^+_i[j] = X^+_{i'}[j']] \leq \frac{q_e^2}{2^n}$$

Therefore,

$$Pr[B5] \leq \frac{(\sigma_e - q_e)\sigma_e + q_e^2}{2^n} \leq \frac{2\sigma_e^2}{2^n}.$$

6. **Pr[B6$\wedge$B1^c]** For $i \in [1 \cdots q_v]$ and $l^+_{i'} > j' > 0$, we need to find the probability of the equality event $X^-_i[p_i + 1] = X^+_{i'}[j']$. The event B1^c implies that there are at most n many possible values for (i', j'). So if we fix (i', j') then we need to bound the probability for equality for the rest $n/2$ bits.
Case(i): $p_i = -1$.
Then we have the equality event as $X^-_i[0] = X^+_{i'}[j']$. Since $p_i = -1$, $N^-_i \neq N^+_{i'}, \forall i' \in [1 \cdots q_e]$. Hence,

$$Pr[X^-_i[0] = X^+_{i'}[j'] \wedge B1^c] \leq \frac{nq_v}{2^{n/2}}$$

(Note that $\Delta^+_{i'}$ is uniformly distributed.)
Case(ii): $0 \leq p_i < l^-_i - 1$.
Since $p_i \geq 0$, we have $N^-_i = N^+_k$ for some k^{th} encryption query. Suppose $k \neq i'$. Then we obtain a non-trivial linear equation on $\Delta^+_{i'}$. Therefore, the probability in this case is at most $\frac{nq_v}{2^{n/2}}$.
Suppose $k = i'$. Then we must have $j' \neq p_i + 1$. Otherwise we get $C^-_i[p_i] = C^+_k[p_i]$ which contradicts the definition of p_i. Hence we get the probability at most $\frac{q_v}{2^{n/2}}$.
Case(iii): $p_i = l^-_i - 1$.

$$Pr[X^-_i[l^-_i] = X^+_{i'}[j'] \wedge B1^c] \leq \frac{nq_v}{2^{n/2}}$$

In all of the above cases, we get the probability at most $\frac{nq_v}{2^{n/2}}$. Hence,

$$Pr[B6 \wedge B1^c] \leq \frac{nq_v}{2^{n/2}}$$

7. **Pr[B7∧ B1^c∧ B2^c]**.
   Case(i): $p_i = -1$.
   Here, $\Delta_{i'}^+$ and $\lceil Y_{i'}^+[l_{i'}^+ - 1]\rceil$ are independent and uniformly distributed. Also i and i' can take at most q_v and c many values respectively. Then

$$Pr[X_i^-[0] = X_{i'}^+[l_{i'}^+] \wedge B2^c] \leq \frac{c\, q_v}{2^{3n/4}} \leq \frac{max\{n, nq_e/2^{n/4}\}q_v}{2^{3n/4}}$$

   .
   Case(ii): $0 \leq p_i < l_i^- - 1$.
   Suppose $N_i^- \neq N_{i'}^+$. Then the equality event $X_i^-[p_i + 1] = X_{i'}^+[l_{i'}^+]$ gives the two $n/2$-bit non-trivial equations, one is in Δ_i^- and another is in $\Delta_{i'}^+$. The event B1^c implies that i' can take at most n many values. Hence the probability of this event is at most $\frac{nq_v}{2^{n/2}}$.
   Suppose $N_i^- = N_{i'}^+$. Then we must have $p_i + 1 \neq l_{i'}^+$ (otherwise we get contradiction to the definition of p_i).Therefore, the equality event $X_i^-[p_i + 1] = X_{i'}^+[l_{i'}^+]$ gives a non-trivial equation in $\Delta_{i'}^+$. Then the probability is at most $\frac{q_v}{2^{n/2}}$.

   Case(iii): $p_i = l_i^- - 1$, we get the same probability bound as in Case(ii). Hence,

$$Pr[B7 \wedge B1^c \wedge B2^c] \leq \frac{max\{n, nq_e/2^{n/4}\}q_v}{2^{3n/4}} + \frac{nq_v}{2^{n/2}}$$

8. **Pr[B8∧B1^c]**.Fix $i \in [1 \cdots q_v]$.
   Since $p_i \geq 0$, we have $N_i^- = N_k^+$ for some k^{th} encryption query. Then for a fixed i,

$$Pr[X_i^-[p_i{+}1] = X_{i_1}^+[0]] = Pr[\lfloor Y_k^+[p_i]\rfloor \oplus \lfloor C_i^-[p_i]\rfloor \oplus Const_{p_i+1} \odot \Delta_k^+ = \lfloor N_{i_1}^+\rfloor_{n/4} || 0^{n-r-2} || b_{1i_1}^+ b_{0i_1}^+]$$

   Then we can bound this event with probability $1/2^{n/4}$.
   Also the event B1^c implies that (i', j') can take at most n many values.

$$Pr[X_i^-[p_i + 2] = X_{i'}^+[j'] \wedge B1^c] \leq \frac{n}{2^{n/2}}$$

   Hence,

$$Pr[B8 \wedge B1^c] \leq \frac{nq_v}{2^{3n/4}}$$

By adding all these probabilities we prove the lemma. □

3.4 Lower Bound of $ip_{real}(\tau)$

We need to find out the lower bound for the ratio of ip_{real} and ip_{ideal}. For that, we fix $\tau \in \mathcal{V}_{good}$, where $\tau = (\tau_e, \tau_v)$ and

$$\tau_e = (N_i^+, A_i^+, M_i^+, X_i^+, Y_i^+, T_i^+)_{i=1..q_e},$$
$$\tau_v = (N_i^-, A_i^-, C_i^-, T_i^-, \perp)_{i=1..q_v}$$

. We assume that all the probability space (except for $ip_{ideal}(*)$) are defined over the real game. Clearly, $ip_{ideal}(\tau) = \frac{1}{2^{n(\sigma_e+q_e)}}$. Now we consider the real case. As **B3, B4, B5** do not hold for this good transcript τ, all the inputs of the random function inside τ_e are distinct and hence all the Y^+-values are independent and uniformly distributed. Also the X^+-values are uniquely determined from Y^+, A^+ and M^+.
Therefore, $\Pr[\tau_e] = \frac{1}{2^{n(\sigma_e+q_e)}}$. Now we have to calculate

$$\begin{aligned} ip_{real}(\tau) &= Pr[\tau_e, \tau_v] \\ &= Pr[\tau_v|\tau_e]Pr[\tau_e] \\ &= \frac{1}{2^{n(\sigma_e+q_e)}} Pr[\tau_v|\tau_e]. \end{aligned} \tag{1}$$

Let η be the event that $\forall i \in [1 \cdots q_v]$, $X_i^-[j]$ for $p_i < j \leq l_i^-$ can not collide with any X^+-values in τ_e and $X_i^-[j]$'for $j \neq j'$. As the events **B6, B7, B8** can not hold for the good transcript, $Y_i^-[p_i+1]$ is uniformly random. Due to the property of feedback function, $X_i^-[p_i+2]$ is also uniformly random.
Now we need to calculate $Pr[\tau_v|\tau_e]$.

$$\begin{aligned} Pr[\tau_v|\tau_e] &= 1 - Pr[\tau_v^c|\tau_e] \\ &= 1 - (Pr[\tau_v^c, \eta|\tau_e] + Pr[\tau_v^c, \eta^c|\tau_e]) \end{aligned} \tag{2}$$

Here, $Pr[\tau_v^c, \eta|\tau_e]$ is the probability that $\exists i \in [1 \cdots q_v]$ such that T_i^- is correct. But $T_i^- = Y_i^-[l_i^-]$ and the event η implies that $Y_i^-[l_i^-]$ is uniformly random. Hence $Pr[\tau_v^c, \eta|\tau_e]$ is the probability of guessing T_i^- correctly.
Therefore,

$$Pr[\tau_v^c, \eta|\tau_e] \leq \frac{q_v}{2^n} \tag{3}$$

Now, $Pr[\tau_v^c, \eta^c|\tau_e]$ and the event η^c can be described as
for $i \in [1 \cdots q_v]$ and $p_i + 1 \leq j \leq l_i^-$, $X_i^-[j] = X_{i_1}^+[j_1] \vee X_i^-[j] = X_i^-[j']$ for some $i_1, j_1, j' \neq j$ and $j' > p_i$.
The event $\mathrm{B1}^c$ can not hold for the good transcript. Hence (i_1, j_1) can take at most n many values. Then for a fixed i, we have
$Pr[X_i^-[j] = X_{i_1}^+[j_1]] \leq \frac{n.l_i^-}{2^{n/2}}$ and $Pr[X_i^-[j] = X_i^-[j'] \wedge T_i^-$ is correct$] \leq \frac{(l_i^-)^2}{2^{n/2}} \frac{1}{2^{n/2}}$. Also $\sum_{1 \leq i \leq q_v}(l_i^-) \leq \sigma_v$ and $\sum_{1 \leq i \leq q_v}(l_i^-)^2 \leq \sigma_v^2$.

Therefore,

$$Pr[\tau_v^c, \eta^c|\tau_e] \leq \frac{n\sigma_v}{2^{n/2}} + \frac{\sigma_v^2}{2^n} \leq \frac{2n\sigma_v}{2^{n/2}} \tag{4}$$

Combining (5), (6), (7) and (8), we get

$$\begin{aligned} ip_{real}[\tau] &\geq \frac{1}{2^{n(\sigma_e+q_e)}}\left(1 - \frac{q_v}{2^n} - \frac{2n\sigma_v}{2^{n/2}}\right) \\ &\geq ip_{ideal}[\tau]\left(1 - \frac{q_v}{2^n} - \frac{2n\sigma_v}{2^{n/2}}\right). \end{aligned}$$

The result follows from Coefficients-H technique combined with Lemma 1. □

References

[1] Ricardo A. Baeza-Yates and Gaston H. Gonnet. *Handbook of Algorithms and Data Structures in Pascal and C*. Addison-Wesley, 1991.

[2] Subhadeep Banik, Andrey Bogdanov, Atul Luykx, and Elmar Tischhauser. Sundae: Small universal deterministic authenticated encryption for the internet of things. *IACR Transactions on Symmetric Cryptology*, 2018(3):1–35, Sep. 2018.

[3] Subhadeep Banik, Sumit Kumar Pandey, Thomas Peyrin, Yu Sasaki, Siang Meng Sim, and Yosuke Todo. GIFT: A small present - towards reaching the limit of lightweight encryption. In Fischer and Homma [6], pages 321–345.

[4] Subhadeep Banik, Sumit Kumar Pandey, Thomas Peyrin, Siang Meng Sim, Yosuke Todo, and Yu Sasaki. GIFT: A small present. *IACR Cryptology ePrint Archive*, 2017:622, 2017.

[5] Avik Chakraborti, Nilanjan Datta, Ashwin Jha, and Mridul Nandi. HYENA. Submission to NIST Lightweight Competition, 2019. https://csrc.nist.gov/CSRC/media/Projects/Lightweight-Cryptography/documents/round-1/spec-doc/hyena-spec.pdf.

[6] Wieland Fischer and Naofumi Homma, editors. *Cryptographic Hardware and Embedded Systems - CHES 2017 - 19th International Conference, Taipei, Taiwan, September 25-28, 2017, Proceedings*, volume 10529 of *Lecture Notes in Computer Science*. Springer, 2017.

[7] Gaston H. Gonnet. Expected length of the longest probe sequence in hash code searching. *J. ACM*, 28(2):289–304, 1981.