HyENA-v2: Tweak Proposal for HyENA

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

[1] Indian Statistical Institute, Kolkata, India
{avikchkrbrti,ashwin.jha1991,mridul.nandi}@gmail.com
[2] Institute for Advancing Intelligence, TCG CREST, Kolkata, India
nilanjan_isi_jrf@yahoo.com

1 HyENA-v2 Authenticated Encryption Mode

In this section, we present the updated formal specification of HyENA-v2 in Algorithm 2. A pictorial description is given in Figure 1. The HyENA authenticated 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 $T \in \{0,1\}^n$.

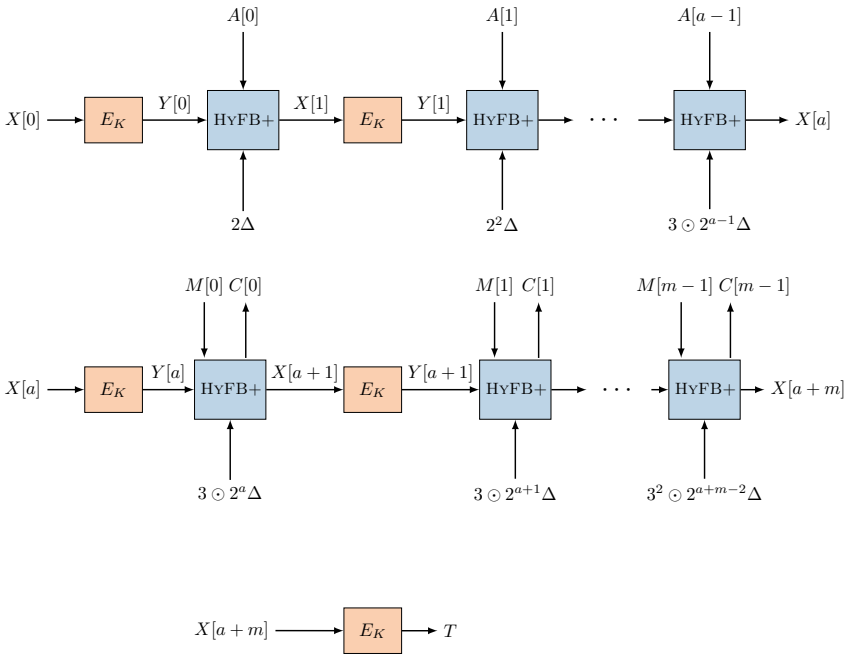

Figure 1: HyENA authenticated encryption mode for full data blocks.

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 return the plaintext $M \in \{0,1\}^{|C|}$, corresponding to the ciphertext C, if the tag T authenticates. Complete specification of HyENA is presented in Algorithm 2 and the corresponding pictorial description can be found in Figure 1.

1.1 The Modifications and Rationale

Here we explicitly mention all the modifications along with the rationale of the modification:

- Updation in the Masking: In the submitted version, for the final block of associated data, the masking value was updated by multipling 2^2 (full block) and 2^3 (partial block). In the current version, we update that by 3 (full block) and 3^2 (partial block). This ensures identical mask generation for associated data and plain text, and makes the associated data and plain text processing uniform.

- Initialization: In the previous version, we differentiate the following cases (i) empty data (both associated data and plain text), (ii) empty associated data and non-empty data, (iii) non-empty associated data by keeping 2 dedicated bits in the initial state. Here we skip this, and we do not separate these cases. The modified padding and the uniqueness of the nonce takes care of it. This saves a MUX in the hardware, and reduces the overall implementation area.

- Finalization: In the previous version, the tag generation was done by (i) swapping the most significant and least significant $n/2$ bits of the state, and then (ii) performing a block cipher encryption. In the updated version, we skip the swap operation, and only perform the encryption. The updated masking takes care of it, and the swap operation becomes redundant.

2 Security

We claim that HyENA-v2 will provide the following security bound:

Theorem 1.

$$\begin{aligned}\mathbf{Adv}^{\mathrm{AE}}_{\text{HyENA-v2}}(q_e, q_v, \sigma_e, \sigma_v, t) \leq & \mathbf{Adv}^{\mathrm{prp}}_{E_K}(q', t') + \frac{2\sigma_e}{2^{n/2}} + \frac{\sigma_e^2 + q_v}{2^n} + \frac{max\{n, nq_e/2^{n/4}\}}{2^{n/4}} \\ & + \frac{n(q_e + 3q_v + 2\sigma_v)}{2^{n/2}} + \frac{max\{n, nq_e/2^{n/4}\}(q_e + 3q_v)}{2^{3n/4}},\end{aligned}$$

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

Theorem 1 follows from [Theorem 1, [1]]. As per the proof, B2 is used specifically to handle the events B4 and B7: specifically, encryption/decryption first input block (where nonce is processed) matches with some encryption last input block (where swapping is used). However, we have found out that the bad case B2 is redundant, and the analysis of B4 and B7 can be bounded given negation of B1 itself.

Applying the above result, we summarize the security clains for HyENA-v2 in Table 1.

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

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

Algorithm Init(N, A, M)

1. $Y \quad E_K(N \| 0^{n-r})$
2. **return** Y

Algorithm Proc-AD(Y, A)

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

Algorithm Tag-Gen(X)

1. $T \quad E_K(X)$
2. **return** T

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

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

Algorithm HyFB+(Y, Δ, M)

1. $C \quad \mathsf{Trunc}_{|M|}(Y) \oplus M$
2. $\overline{M} \quad \mathsf{Pad}(M),\ \overline{C} \quad \mathsf{Pad}(C)$
3. $B \quad \left(\overline{M}_L \| (\overline{C}_R \oplus \Delta)\right)$
4. $X \quad B \oplus Y$
5. **return** (X, C)

Algorithm HyFB−(Y, Δ, C)

1. $M \quad \mathsf{Trunc}_{|C|}(Y) \oplus C$
2. $\overline{M} \quad \mathsf{Pad}(M),\ \overline{C} \quad \mathsf{Pad}(C)$
3. $B \quad \left(\overline{M}_L \| (\overline{C}_R \oplus \Delta)\right)$
4. $X \quad B \oplus Y$
5. **return** (X, M)

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

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

Figure 2: Formal Specification of HyENA Authenticated Encryption and Decryption algorithm. We use the notation $\star$ to denote values that we do not care.

Table 1: Summary of security claims for HyENA. The data and time limits indicate the amount of data and time required to make the attack advantage close to 1.

Submissions	**Privacy**		**Integrity**	
	Time	Data (in bytes)	Time	Data (in bytes)
HyENA-v2	2^{128}	2^{64}	2^{128}	2^{58}

Table 2: Clock cycles per message byte for HyENA-v2

	Message length (Bytes)											
	16	32	64	128	256	512	1024	2048	4096	16384	32768	262144
cpb	10.563	6.656	4.7031	3.727	3.238	2.9941	2.872	2.811	2.781	2.758	2.754	2.750

2.1 Third Party Analysis of HyENA NIST Submission

To the best of our knowledge, there is one valid third-party analysis [2] on HyENA NIST submitted version. This attack actually targets the Δ update part and exploits the fact we have a collision in the Δ value corresponding to two different block processing. Finally, the attack use the collision to make a valid forgery. Due to this attack HyENA needs a minor fix to achieve a stong mathematical property in Δ update. This writeup makes such minor update in the Δ update function to obtain a prefix-free in the (i, j) sequence used for Δ update as $2^i \cdot 3^j \Delta$. This strong mathematical makes the mode provably secure.

3 Hardware Implementation of HyENA-v2

HyENA-v2 aims to achieve a lightweight implementation on low resource devices. We have already presented the hardware implementation results of a close variant of the NIST submission HyENA in [1] (the NIST and this version differs only in the Δ update). The implementation details in [1] is also presented in this write-up in Table 3. HyENA-v2 further simplifies the structure with the modifications described in Sect 1.1 (in fact, HyENA-v2 omits two operations). Hence, HyENA-v2 is expected to have better hardware implementation results than the results reported in [1]. The hardware benchmarking results given in [1, Table 4], also show that HyENA significantly outperforms other well established ciphers such as AES-GCM, ASCON, AES-OTR etc in terms of area. This establishes HyENA as one of the best ultra-lightweight AEAD cipher.

3.1 Clock Cycle Analysis

We provide a conventional way for speed estimation, i.e, the number of clock cycles to process input bytes. Since HyENA-v2 processes at least one associated data (AD) block (one dummy block when AD is empty), we calculate the cpb assuming one AD block and m message blocks. We use 40 round GIFT and need 40 cycles for the GIFT module. We use 4 more cycles to compute the feedback and update the Δ value. Overall, HyENA needs $(44(m+1)+81)$ cycles. Table 2 shows the number of average cycles per input message bytes, which we call cycles per byte (cpb). The cpb is $(44(m+1)+81)/16m$ and it converges to 2.75 for very large m.

3.2 Implementation Results

We implement HyENA-v2 on Xilinx Virtex 6 and Virtex 7, using VHDL and Xilinx ISE 13.4. Table 3 presents the implementation results of HyENA-v2 on Virtex 7.We follow the RTL approach and a basic iterative type architecture with 128-bit datapath. The ares are provided in the number of LUTs and slices. Frequency (MHz), Throughput (Gbps), and

Table 3: FPGA implementation results of HyENA

Design (Platform)	Slice Registers	LUTs	Slices	Frequency (MHz)	Throughput (Gbps)	Mbps/LUT	Mbps/Slice
HyENA-v2 (Virtex 7)	336	668	322	410.295	1.28	1.916	3.975

throughput-area efficiencies are also reported in addition to the hardware areas. Table 3 presents the mapped hardware results of HyENA.

References

[1] Avik Chakraborti, Nilanjan Datta, Ashwin Jha, Snehal Mitragotri, and Mridul Nandi. From combined to hybrid: Making feedback-based AE even smaller. *IACR Trans. Symmetric Cryptol.*, 2020(S1):417–445, 2020.

[2] Alexandre Mege. ROUND 2 OFFICIAL COMMENT - HYENA, 2020. https://csrc.nist.gov/CSRC/media/Projects/lightweight-cryptography/documents/round-2/official-comments/hyena-round2-official-comment.pdf.