SUNDAE: Small Universal Deterministic Authenticated Encryption for the IoT

Subhadeep Banik\textsuperscript{1,4}, Andrey Bogdanov\textsuperscript{2}, Atul Luykx\textsuperscript{3}, Elmar Tischhauser\textsuperscript{2}

\textsuperscript{1}LASEC, EPFL Switzerland
\textsuperscript{2}Technical University of Denmark, Lyngby
\textsuperscript{3}Visa Research, USA
\textsuperscript{4}NTU, Singapore

Fast Software Encryption 2019, Paris
25th March 2018
Outline

- Introduction
- Specification
- Security
- Implementation
Introduction

Block Cipher based AE

- Block cipher is an efficient component for lightweight AE.

- SIV (Eurocrypt 2006) mode requires 2 independent keys.

- Some candidates:
  - COPA/EℓMD/COLM: Internal state size at least 3 times of block length.
  - EAX: Multiple initial block cipher calls.
  - COFB/JAMBU: State size greater than block length.

- GCM-SIV proposed at CCS 2015.
  - Multiplication in $GF(2^{128})$: not efficient in hardware.
Contributions

SUNDAE

- Competes with CLOC/JAMBU in number of block cipher calls for short messages
- Improves COFB and other modes in terms of state size
- Simultaneously offers efficiency on lightweight and high-performance platforms
- Provides maximal robustness to a lack of proper randomness
SUNDAE

- Completely deterministic:
  → If input is unique, it maintains both data confidentiality and authenticity.

- Processes inputs of the form $(A, M)$
  → If $M$ is empty, the mode reduces to a MAC.
  → If nonce is required, the first $x$ bits of $A$ can serve the purpose.

- Structure is based on SIV, optimized for lightweight settings:
  → Uses one key, consists of a cascade of block cipher calls.
  → Only additional operations: XOR and multiplication by fixed constants.

- State size of $n$, where $n$ is blocklength of underlying block cipher.
  → CLOC requires $2n$-bits, JAMBU $1.5n$-bits, and COFB $1.5n$-bits.
Characteristics

SUNDAE

- Rate 1/2 mode:
  → 2 block cipher calls per message block.

- Efficient for short messages: for 1 block of nonce, plaintext, AD
  → COFB uses 3 block cipher calls, CLOC requires 4, JAMBU 5.
  → SUNDAE requires 5 calls (can be reduced to 4, if one call is precomputed).

- Hence efficient in settings where communication outweighs computational costs
  → If AD/plaintext is never repeated,
  → nonce is no longer needed, and
  → communication or synchronization costs are reduced,
  → in addition to reducing the block cipher calls to 4
Specification

Algorithm 1: $\text{enc}_K(A, M)$

Input: $K \in K, A \in \{0, 1\}^*, M \in \{0, 1\}^*$

Output: $C \in \{0, 1\}^{n+|M|}$

1. $b_1 \leftarrow |A| > 0 \ ? 1 : 0$
2. $b_2 \leftarrow |M| > 0 \ ? 1 : 0$
3. $V \leftarrow E_K(b_1 || b_2 || 0^{n-2})$
4. $T \leftarrow V$ // Initial tag
5. if $|A| > 0$ then
   7. for $i = 1$ to $\ell_A - 1$ do
      8. $V \leftarrow E_K(V \oplus A[i])$
   9. end
10. $X \leftarrow |A[\ell_A]| < n \ ? 2 : 4$
11. $V \leftarrow E_K(X \times (V \oplus \text{pad}(A[\ell_A])))$
12. $T \leftarrow V$
13. end
14. if $|M| > 0$ then
   16. for $i = 1$ to $\ell_M - 1$ do
      17. $V \leftarrow E_K(V \oplus M[i])$
   18. end
   19. $X \leftarrow |M[\ell_M]| < n \ ? 2 : 4$
   20. $V \leftarrow E_K(X \times (V \oplus \text{pad}(M[\ell_M])))$
   21. $T \leftarrow V$
   22. for $i = 1$ to $\ell_M$ do
      23. $V \leftarrow E_K(V)$
      24. $C[i] \leftarrow [V | M[i]| \oplus M[i]$
   25. end
   26. return $TC[1] \cdots C[\ell_M]$
27. end
28. return $T$
Algorithm 2: $\text{dec}_K(A, C)$

Input: $K \in K$, $A \in \{0, 1\}^*$, $C \in \{0, 1\}^n \times \{0, 1\}^*$

Output: $\bot$ or $M \in \{0, 1\}^{|C| - n}$

2. $V \leftarrow C[1]$
3. for $i = 2$ to $\ell$ do
   4. $V \leftarrow E_K(V)$
   5. $M[i - 1] \leftarrow [V]_{|M[i]|} \oplus C[i]$
4. end
6. $T \leftarrow [\text{enc}_K(A, M)]_n$
7. if $T \neq C[1]$ then
6. return $\bot$
4. return $M$
Specifications

Figure: SUNDAE encryption with associated and plaintext data. The box below the rightmost block cipher call represents truncation.

9 Subhadeep Banik  SUNDAE: Small Universal Deterministic Authenticated Encryption for the IoT  25.3.2019
Figure: SUNDAE encryption with associated and plaintext data. The box below the rightmost block cipher call represents truncation.
Theorem

Let $A$ be an adversary making at most $q_{\text{enc}}_K$ and $q_v$ $\text{dec}_K$ queries with block length costs of at most $\sigma_A$, $\sigma_P$, and $\sigma_C$ for associated, plaintext, and ciphertext data, respectively, then

$$\text{DAE}(A) \leq \frac{N_E^2}{2^{n+1}} + \frac{q_v}{2^n} + \frac{q^2}{2^n} + \frac{qq_v}{2^n} + \frac{(\sigma_P + \sigma_C)^2}{2^{n+1}} + \frac{4(\sigma_P + \sigma_C)}{2^n} +$$

$$\frac{(4 + \sigma_A + \sigma_P + \sigma_C)^2}{2^n} + \frac{4(q + q_v)^2}{2^n} + \text{PRP}_E(A_E).$$  \hspace{1cm} (1)

where

$$N_E := 4 + \sigma_A + 2\sigma_P + 2\sigma_C$$  \hspace{1cm} (2)
Proof Intuition: Step 1 (Switching to URF)

Values of IV

| |AD|,|PT| |AD/PT[ℓ]| |
|---|---|---|---|
|0,0|000\(n-2\)|&lt;n|&lt;2|
|0,1|010\(n-2\)|&lt;n|&lt;2|
|1,0|100\(n-2\)|=n|&times;4|
|1,1|110\(n-2\)|&gt;n|&gt;4|

Constant Multiplication

\[ \Delta (\text{enc}_K, \text{dec}_K; $, \bot) \]

\[ := \Delta (\text{enc}[\rho], \text{dec}[\rho]; $, \bot) + \frac{N_E^2}{2n+1} + \text{PRP}_E(A_E), \]

DAE(A) := \Delta (\text{enc}_K, \text{dec}_K; $, \bot)
Proof Intuition: Step 1 (Switching to URF)

- We use stream cipher OFB, unpredictable SIV \(\rightarrow\) confidentiality.
- Confidentiality will be maintained if the tag is unpredictable.
- AD/PT is processed similarly, we argue that the domain separation works.
Proof Intuition: Step 1 (Authenticity)

- Adversary forges \((C, T) \rightarrow\) output of MAC for \(\text{dec}(C, T)\)- call equals \(T\)
- By defn, \(C\) was never before output of previous enc query.
- Equivalent to producing pre-image/2nd pre-image of underlying MAC.
Proof Intuition: Step 2 (eliminate chopxor)

- $TC = \text{enc}(A,M) = \text{chopxor}_M \circ \text{enc-stream}(A,M)$
- $M' = \text{chopxor}_C \circ \text{stream}(T)$. Compute $T' = 1$st block of $\text{enc-stream}(A,M')$
- If $T = T'$, $\text{dec-stream}(A,TC) = \text{stream}(T)$ else $\bot$.
- $M = \text{dec}(A,TC) = \text{chopxor}_C \circ \text{dec-stream}(A,TC)$
Proof Intuition: Step 2 (eliminate chopxor)

- **DAE(A)** := \(\Delta_A (\text{enc}[\rho], \text{dec}[\rho] ; $, \bot) + \frac{N_E^2}{2^{n+1}} + \text{PRP}_E(A_E)\)
- \(\Delta_A (\text{enc}[\rho], \text{dec}[\rho] ; $, \bot) \leq \Delta_{A_{\text{chopxor}}} (\text{enc-stream, dec-stream} ; $^S, \bot)\)
- Where \$^S returns random string of length \((\ell_M + 1) \ast n\)
Proof Intuition: Step 3 (introduce stream*/decstream*)

- stream*(T) outputs completely random values of required length.
- If $T = T_i$ for some $i$, dec-stream*(A,TC) outputs stream*(T_i) else ⊥

$$\Delta_{A_{\text{chopxor}}} (\text{enc-stream}, \text{dec-stream} ; \$^s, \perp) \leq \Delta_{A_{\text{chopxor}}} (\text{enc-stream}, \text{dec-stream} ; \$^s, \text{dec-stream}^*) + \Delta_{A_{\text{chopxor}}} (~\$^s, \text{dec-stream}^* ; \$^s, \perp)$$
Proof Intuition: Step 3 (introduce stream*/decstream*)

- $\Delta_{A_{\text{chopxor}}} (s^s, \text{dec-stream}^*; s^s, \bot) = \text{prob that decstream}^* \text{ outputs non-} \bot$
- Same as finding pre-image/second pre-image for $\lfloor s^s \rfloor_n$

$$\Delta_{A_{\text{chopxor}}} (s^s, \text{dec-stream}^*; s^s, \bot) \leq \frac{q_v}{2^n} + \frac{q^2}{2^n} + \frac{qq_v}{2^n}.$$  \hspace{1cm} (3)
Proof Intuition: Step 3 (introduce stream*/decstream*)

- Remaining term $\Delta_{A_{\text{chopxor}}} (\text{enc-stream}, \text{dec-stream}; \bar{s}, \text{dec-stream}^*)$
- We will try to bound using H-coefficient technique.
Proof Intuition: Step 4 (message to function)

- Split $A$ and $M$ into blocks, if non-empty, to get
  \[ A[1] \cdots A[\ell_A] \leftarrow^n A \text{ and } M[1] \cdots M[\ell_M] \leftarrow^n M. \]  
  \[ (4) \]

- Each block augmented with a bit to indicate if it is a final block or not.
  \[ \left( (0, A[1]), \ldots, (1, A[\ell_A]), (0, M[1]), \ldots, (1, M[\ell_M]) \right). \]
  \[ (5) \]

- The augmented blocks are used as parameter in the function
  \[ f : \left( \{0, 1\} \times \{0, 1\}^{\le n} \right) \times B \rightarrow B, \]
  \[ (6) \]

  where $f$ is defined as

  \[ f((\delta, X), Y) := \begin{cases} 
  X \oplus Y & \text{if } \delta = 0 \\
  2 \times (\text{pad}(X) \oplus Y) & \text{if } \delta = 1 \text{ and } |X| < n \\
  4 \times (X \oplus Y) & \text{otherwise} 
  \end{cases} \]
  \[ (7) \]
Proof Intuition: Step 4 (message to function)

• If \( A \neq \varepsilon \) and \( M \neq \varepsilon \), we have that \( f((\delta, X), Y) \) and \( f_{\delta, X}(Y) \) are equiv

\[
I(A, M) := \left( 110^{n-2}, f_{0, A[1]}, \ldots, f_{0, A[\ell-1]}, f_{1, A[\ell_A]}, f_{0, M[1]}, \ldots, f_{0, M[\ell-1]}, f_{1, M[\ell_M]} \right), \tag{8}
\]

where values \( X \in \{0, 1\}^n \) are interpreted as constant functions mapping any element in \( B \) to \( X \).

• Given \( \vec{x} = (x_1, x_2, \ldots, x_\ell) \) where each \( x_i \) is a function, define

\[
\hat{\rho}(x_1, x_2, \ldots, x_\ell) = \rho \circ x_\ell \circ \rho \circ x_{\ell-1} \circ \cdots \circ \rho \circ x_3 \circ \rho \circ x_2 \circ \rho \circ x_1. \tag{9}
\]

It is easy to see \( \text{enc-stream}(A, M) := \text{stream}_{\ell_M}(\hat{\rho}(I(A, M))) \)
**Proof Intuition: Step 5 (function to graph)**

- Convert transcript to a graph, respecting prefix rules.
- Output streams exist as independent, unconnected nodes.
- Very natural to transform values to functions.
- Each edge becomes application of $\rho$, each node has label $\chi_i$. 
Proof Intuition: Step 5 (function to graph)

- Define $T_{bad}$ for all transcripts that lead to events 1,2
- Allows trivial forgery.
- Concentrate on $T_{good}$
Proof Intuition: Step 5 (function to graph)

- Structural collision: when two unequal values lead to same function.
- Natural isomorphism between the 2 graphs no longer maintained.
- This can never happen in SUNDAE. Mapping from $\delta, X \rightarrow f_{\delta,X}$ is injective.
The next event is $\rho$-coll$_{\vec{t}}$: if labels of 2 nodes become equal.

May occur due to randomness introduced by the URF $\rho$.

We use graph-theoretic arguments to bound prob of $\rho$-coll$_{\vec{t}}$. 
Proof Intuition: Step 5 (function to graph)

- Now straightforward to apply H-coeffs. Adding we get bound in Thm 1.

\[
\Delta_{A_{chopxor}}(\text{enc-stream, dec-stream}; s^*, \text{dec-stream}^*) \leq \\
\frac{(\sigma_P + \sigma_C)^2}{2^{n+1}} + \frac{4(\sigma_P + \sigma_C)}{2^n} + \frac{(4 + \sigma_A + \sigma_P + \sigma_C)^2}{2^n} + \frac{4(q + q_v)^2}{2^n}. \quad (10)
\]
Performance

Software

- Platforms: Cortex-A57 core of a Samsung Exynos 7420 CPU (ARMv8 platform), Intel Core i7-6700 CPU (Skylake)
- Message lengths: $\ell = 2^b$ bytes, with $6 \leq b \leq 11$, with comb scheduling.

- On Intel, SUNDAE is around 3% slower than two passes of CBC; on ARM, 7%.
- For short messages only around 11% worse than for longer messages.

- Compared to the single-pass COFB, SUNDAE has an overhead of 60% for short and 80% for long messages on Intel
- And 35% for short and 80% for long messages on ARM.
## Performance

Table: ARMv8 platform (embedded)

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>64</th>
<th>128</th>
<th>256</th>
<th>512</th>
<th>1024</th>
<th>2048</th>
<th>mix</th>
</tr>
</thead>
<tbody>
<tr>
<td>CBC (S)</td>
<td>2.69</td>
<td>2.54</td>
<td>2.39</td>
<td>2.30</td>
<td>2.26</td>
<td>2.25</td>
<td>2.38</td>
</tr>
<tr>
<td>CBC (P)</td>
<td>1.42</td>
<td>1.14</td>
<td>1.02</td>
<td>0.95</td>
<td>0.92</td>
<td>0.90</td>
<td>1.00</td>
</tr>
<tr>
<td>COFB (S)</td>
<td>3.99</td>
<td>3.34</td>
<td>2.96</td>
<td>2.78</td>
<td>2.72</td>
<td>2.71</td>
<td>2.98</td>
</tr>
<tr>
<td>COFB (P)</td>
<td>2.98</td>
<td>1.89</td>
<td>1.49</td>
<td>1.32</td>
<td>1.25</td>
<td>1.22</td>
<td>1.52</td>
</tr>
<tr>
<td>SUNDAE (S)</td>
<td>5.42</td>
<td>5.14</td>
<td>5.02</td>
<td>4.92</td>
<td>4.86</td>
<td>4.84</td>
<td>4.97</td>
</tr>
<tr>
<td>SUNDAE (P)</td>
<td>3.16</td>
<td>2.95</td>
<td>2.85</td>
<td>2.80</td>
<td>2.78</td>
<td>2.76</td>
<td>2.84</td>
</tr>
</tbody>
</table>
### Performance

**Table: Intel Skylake platform (server)**

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>64</th>
<th>128</th>
<th>256</th>
<th>512</th>
<th>1024</th>
<th>2048</th>
<th>mix</th>
</tr>
</thead>
<tbody>
<tr>
<td>CBC (S)</td>
<td>2.90</td>
<td>2.75</td>
<td>2.68</td>
<td>2.63</td>
<td>2.60</td>
<td>2.59</td>
<td>2.67</td>
</tr>
<tr>
<td>CBC (P)</td>
<td>0.64</td>
<td>0.64</td>
<td>0.63</td>
<td>0.63</td>
<td>0.63</td>
<td>0.63</td>
<td>0.64</td>
</tr>
<tr>
<td>COFB (S)</td>
<td>3.71</td>
<td>3.32</td>
<td>3.12</td>
<td>3.02</td>
<td>2.97</td>
<td>2.96</td>
<td>3.12</td>
</tr>
<tr>
<td>COFB (P)</td>
<td>1.03</td>
<td>0.95</td>
<td>0.90</td>
<td>0.87</td>
<td>0.86</td>
<td>0.85</td>
<td>0.90</td>
</tr>
<tr>
<td>SUNDAE (S)</td>
<td>6.00</td>
<td>5.71</td>
<td>5.57</td>
<td>5.46</td>
<td>5.40</td>
<td>5.37</td>
<td>5.52</td>
</tr>
<tr>
<td>SUNDAE (P)</td>
<td>1.36</td>
<td>1.31</td>
<td>1.29</td>
<td>1.27</td>
<td>1.26</td>
<td>1.26</td>
<td>1.28</td>
</tr>
</tbody>
</table>
On ASIC

- Replace $2x$ on $GF(2^{128})$ → eight $2x$ over $GF(2^{16})/ < x^{16} + x^5 + x^3 + x + 1 >$
- If $c_0, c_1, \ldots, c_{15}$ denote the individual bytes
- $i^{th}$ bits of each byte is an element of $GF(2^{16})$
- We have: $f(c_0, \ldots, c_{15}) = c_1, c_2, \ldots, c_{11} \oplus c_0, c_{12}, c_{13} \oplus c_0, c_{14}, c_{15} \oplus c_0, c_0$
• Fits well into the bytewise AES circuit: only few gates required.
• Mapping from $\delta, X \rightarrow f_{\delta,X}$ is still injective.
• No change in security guarantees.
• No additional state needs to be stored/updated.
## Performance On ASIC

<table>
<thead>
<tr>
<th>Mode</th>
<th>Underlying Cipher</th>
<th>Blocksize/Keysize</th>
<th>Area (GE)</th>
<th>Power (µW)</th>
</tr>
</thead>
<tbody>
<tr>
<td>CLOC (A)</td>
<td>AES-128</td>
<td>128/128</td>
<td>3110</td>
<td>131.1</td>
</tr>
<tr>
<td>CLOC (C)</td>
<td>AES-128</td>
<td>128/128</td>
<td>4310</td>
<td>156.6</td>
</tr>
<tr>
<td>SILC (A)</td>
<td>AES-128</td>
<td>128/128</td>
<td>3110</td>
<td>131.0</td>
</tr>
<tr>
<td>SILC (C)</td>
<td>AES-128</td>
<td>128/128</td>
<td>4220</td>
<td>155.6</td>
</tr>
<tr>
<td>AES-OTR (A)</td>
<td>AES-128</td>
<td>128/128</td>
<td>4720</td>
<td>164.3</td>
</tr>
<tr>
<td>AES-OTR (C)</td>
<td>AES-128</td>
<td>128/128</td>
<td>6770</td>
<td>205.4</td>
</tr>
<tr>
<td>AES-SUNDAE</td>
<td>AES-128</td>
<td>128/128</td>
<td>2524</td>
<td>126.1</td>
</tr>
<tr>
<td>Present-SUNDAE</td>
<td>Present</td>
<td>64/80</td>
<td>1452</td>
<td>50.9</td>
</tr>
</tbody>
</table>

Table: Implementation results for CLOC, SILC, AES-OTR, and SUNDAE. (Power reported at 10 MHz, A: Aggressive, C: Conservative)
THANK YOU