SEA for internet-0: a Scalable Encryption Algorithm for Small Embedded Applications

F.-X. Standaert, G. Piret, N. Gershenfeld, J.-J. Quisquater
why we need crypto for internet-0

- **identification**: use encryption in a specific way
- **authentication**: use encryption in a specific way
- **encryption**: indeed

- thus asking encryption algorithm with *security*, *scalability* (uses, processors, …), *small footprint*, maybe trading it.
The competition

- Triple-DES:
- AES:
- TEA (Tiny Encryption Algorithm), FSE 1994
- Yuval: “Reinventing the Travois”, FSE 1997
  - no scalability
  - do we only need one standard encryption algorithm?
    (NB: need of something like tinySSL for UDP)
Specifications

- $n$: plaintext size, key size.
- $b$: processor (or word) size.
- $n_b = \frac{n}{2b}$: number of words per Feistel branch.
- $n_r$: number of block cipher rounds.

$\rightarrow \text{SEA}_{n,b}$
Basic operations

1. Bitwise XOR $\oplus$
2. Substitution box $S$
3. Word rotation $R$
4. Bit rotation $r$
5. Addition mod $2^b$ $\boxplus$
Round and key round

\[ \begin{aligned}
&L_i \\
&\downarrow R \\
&\downarrow R^{-1} \\
&L_{i+1} \\
\end{aligned}
\quad
\begin{aligned}
&R_i \\
&\downarrow K_i \\
&\downarrow \text{S} \\
&K_{i+1} \\
\end{aligned}
\quad
\begin{aligned}
&K_{L_i} \\
&\downarrow r \\
&\downarrow S \\
&K_{R_i} \\
\end{aligned}

\begin{aligned}
&\downarrow C_i \\
&\downarrow R \\
&\downarrow r \\
&\downarrow \text{S} \\
&\downarrow K_{R_i+1} \\
\end{aligned}
Performances

- Pseudo-assembly code provided
- Operation counts easy
- Only a few instructions required:
  - Arithmetic and logic
  - Branch instructions
  - Comparisons, load from RAM, store in RAM
It yields…

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>$E/D$</th>
<th>Device</th>
<th># ram</th>
<th># regs.</th>
<th>code size (ops.)</th>
<th>implementation time (ops.)</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\text{SEA}_{n,b}$</td>
<td>$4n_b$</td>
<td>$n_b + 3$</td>
<td>31$n_b$+36</td>
<td>$(n_T - 1) \times (22n_b + 29) + 20n_b + 18$</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>$E/D$</th>
<th>Device</th>
<th># ram</th>
<th># regs.</th>
<th>code size</th>
<th># clock cycles</th>
<th># cycles $\times$ code size</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\text{SEA}_{96,8}$</td>
<td>yes</td>
<td>Atmel ATtiny</td>
<td>1</td>
<td>32</td>
<td>386</td>
<td>17 745</td>
<td>6849.10$^3$</td>
</tr>
<tr>
<td>$\text{SEA}_{192,32}$</td>
<td>yes</td>
<td>ARM (risc-32)</td>
<td>6</td>
<td>12</td>
<td>420</td>
<td>27 059</td>
<td>11 364.10$^3$</td>
</tr>
<tr>
<td>$\text{Rijndael [19]}$</td>
<td>no</td>
<td>ARM (risc-32)</td>
<td>16</td>
<td>12</td>
<td>1404</td>
<td>2889</td>
<td>4056.10$^3$</td>
</tr>
<tr>
<td>$\text{SEA}_{128,32}$</td>
<td>yes</td>
<td>ARM (risc-32)</td>
<td>6</td>
<td>12</td>
<td>280</td>
<td>18 039</td>
<td>5050.10$^3$</td>
</tr>
</tbody>
</table>

**Table 1.** Performance evaluation of $\text{SEA}_{n,b}$ (encryption + decryption).

**Table 2.** Comparisons: the code size is expressed in bytes. The results of $\text{SEA}_{128,32}$ where obtained by multiplying the code size and number of cycles of $\text{SEA}_{192,32}$ by 2/3, since 128 is not a multiple of 6.
ASIC and FPGA

- ASIC: 6800 gates 271 Mbits/s (250 MHz)
- FPGA: 400 slices 240 Mbit/s (240 MHz)
Conclusions

- Efficient combination of encryption/decryption
- Low code size
- Low memory requirements (RAM + regs)
- Typical performances: a few milliseconds and a few hundreds bytes of ROM
- More efficient for large bus sizes
- Compared to the AES: trades time for space
  - May be reasonable in recent controllers