FELICS-AE: a framework to benchmark lightweight authenticated block ciphers

P. Huynh(1), K. Le Gouguec(2)

(1)LORIA CNRS, (2)Airbus CyberSecurity

November 4
2019
• **Background:** the FELICS framework

• **FELICS-AE additions**
  - Support for `crypto_aead`-compliant algorithms
  - New scripts to run benchmarks & analyze results
  - Docker image
  - More platforms

• **Results with Lilliput-AE, Ascon and ACORN**

• **Future work**

• **Questions**
Background: the FELICS framework

FELICS-AE additions

- Support for crypto_aead-compliant algorithms
- New scripts to run benchmarks & analyze results
- Docker image
- More platforms

Results with Lilliput-AE, Ascon and ACORN

Future work

Questions
The original FELICS framework

https://www.cryptolux.org/index.php/FELICS

Benchmarking framework for crypto algorithms focused on small platforms

- Developed by the CryptoLUX research group at University of Luxembourg
- Focuses on block ciphers and stream ciphers
- Benchmark results showcased on the CryptoLUX wiki
Platforms:

- 8-bit AVR ATmega128 (*simulated*)
- 16-bit MSP430F1611 (*simulated*)
- 32-bit ARM Cortex-M3 (*requires Arduino Due and J-Link probe*)
Platforms:

- 8-bit AVR ATmega128 (*simulated*)
- 16-bit MSP430F1611 (*simulated*)
- 32-bit ARM Cortex-M3 (*requires Arduino Due and J-Link probe*)

Metrics:

- Binary code size
- RAM footprint
- Execution time
Background: the FELICS framework

FELICS-AE additions

- Support for crypto_aead-compliant algorithms
- New scripts to run benchmarks & analyze results
- Docker image
- More platforms

Results with Lilliput-AE, Ascon and ACORN

Future work

Questions
Started off FELICS v1.1.0 to compare LILLIPUT-AE to CAESAR’s final lightweight portfolio (use-case 1)

https://gitlab.inria.fr/minier/felics-ae
crypto_aead-compliant algorithms

Minimize work needed to add algorithms that comply with the API

Not quite “drop-in” yet: still some idiosyncrasies from original FELICS

- `ROM_DATA.../RAM_DATA...` macros to specify alignment & storage
- split files for encryption/decryption/common code
felics-run

Orchestrates original FELICS entry points (scripts & makefiles)

Single output format (JSON) used by other scripts
felics-run example

$ ./scripts/felics-run -a PC Lilliput-II-128_vfelicsref
[...]
On PC
Lilliput-II-128 (felicsref, -O3): 6904 528 16792
$ cat results/$date-$time-master.json
{
    "commit": "ef3c770",
    "branch": "master",
    "data": [
        {
            "cipher_name": "Lilliput-II-128",
            "architecture": "PC",
            "version": "felicsref",
            "compiler_options": "-O3",
            "code_size": 6904,
            "code_ram": 528,
            "code_time": 16792
        }
    ]
}
felics-compare

$ ./scripts/felics-compare old.json new.json
Comparing
    old.json
    (master) 1234567 Old commit summary
against
    new.json
    (master) 89abcde New commit summary

Lilliput-I-128 on AVR (vfelicsref with -Os)
    code_size: \textbf{-12.19\%} (3166 \downarrow 2780)
    code_ram: \textbf{-49.22\%} (514 \downarrow 261)
    code_time: \textbf{+32.05\%} (189818 \uparrow 250657)

Lilliput-I-192 on AVR (vfelicsref with -Os)
    code_size: \textbf{-10.47\%} (3268 \downarrow 2926)
    code_ram: \textbf{-50.71\%} (562 \downarrow 277)
    code_time: \textbf{+36.73\%} (230309 \uparrow 314893)
felics-publish

$ ./scripts/felics-publish foo.json -o foo.$format

Options:

--sort-by: how setups are ordered
--filter: which setups are included
--info: which metadata and metrics are displayed
--table-label: anchor for documents supporting cross-references
--table-caption: additional text to describe the data set
felics-publish examples

To \LaTeX: 

\begin{table}
\begin{tabular}{|c|c|c|c|c|c|}
\hline
Version & CFLAGS & Code size (B) & RAM (B) & Execution time (cycles) \\
\hline
Lilliput-I-128 & felicsref & -03 & 6100 & 266 & 129093 \\
Lilliput-II-128 & felicsref & -03 & 6062 & 243 & 132650 \\
Lilliput-I-128 & felicsref & -0s & 2780 & 261 & 250657 \\
Lilliput-II-128 & felicsref & -0s & 2768 & 229 & 297992 \\
\hline
\end{tabular}
\end{table}

**Table:** Performance results for 128-bit key algorithms on AVR ATmega128.

To spreadsheet:
FELICS dependencies

Lots of dependencies to manage:

<table>
<thead>
<tr>
<th>Platform</th>
<th>Software</th>
<th>Origin</th>
</tr>
</thead>
<tbody>
<tr>
<td>AVR</td>
<td>simavr</td>
<td>GitHub</td>
</tr>
<tr>
<td></td>
<td>Avrora</td>
<td>SourceForge + CryptoLUX patch</td>
</tr>
<tr>
<td>MSP</td>
<td>MSP430-GCC</td>
<td>Texas Instruments</td>
</tr>
<tr>
<td></td>
<td>MSPDebug</td>
<td>GitHub</td>
</tr>
<tr>
<td>ARM</td>
<td>J-Link Software</td>
<td>SEGGER</td>
</tr>
</tbody>
</table>

Table: Extra-distro dependencies.

How to distribute such a framework?
Distribution & setup

Original FELICS solutions:

- Documentation\(^1\)
- Virtual machine

\(^1\)https://www.cryptolux.org/index.php/FELICS_Prerequisites
Distribution & setup

Original FELICS solutions:

- Documentation\(^1\)
- Virtual machine

FELICS-AE additions:

- Script to fetch & install all dependencies
- Scripts to generate & run Docker image

---

\(^1\)https://www.cryptolux.org/index.php/FELICS_Prerequisites
New platforms

Platforms:

- 8-bit AVR ATmega128
- 16-bit MSP430F1611
- 32-bit ARM Cortex-M3
New platforms

Platforms:

- 8-bit AVR ATmega128
- 16-bit MSP430F1611
- 32-bit ARM Cortex-M3
- **NEW** 32-bit NRF52840 Cortex-M4
- **NEW** 32-bit STM32L053 Cortex-M0+
Background: the FELICS framework

FELICS-AE additions

- Support for crypto_aead-compliant algorithms
- New scripts to run benchmarks & analyze results
- Docker image
- More platforms

Results with Lilliput-AE, Ascon and ACORN

Future work

Questions
https://paclido.fr/lilliput-ae/implementation/

- **LILLIPUT-AE** on par with or faster than **ASCON** and **ACORN** on 8-bit and 16-bit
- much slower on 32-bit
Background: the FELICS framework

FELICS-AE additions

- Support for crypto_aead-compliant algorithms
- New scripts to run benchmarks & analyze results
- Docker image
- More platforms

Results with Lilliput-AE, Ascon and ACORN

Future work

Questions
Future work

- Integrate more LWC candidates
- Support more than one test vector per algorithm
- Support more scenarios
- documentation/TODO.md
Background: the FELICS framework

FELICS-AE additions

- Support for crypto_aead-compliant algorithms
- New scripts to run benchmarks & analyze results
- Docker image
- More platforms

Results with Lilliput-AE, Ascon and ACORN

Future work

Questions
Any questions?

For more technical inquiries: ask kevin.legouguec@airbus.com!