
Retrofitting Leakage Resilient Authenticated Encryption to Microcontrollers

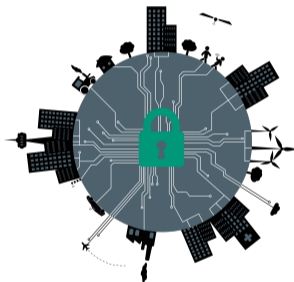
Florian Unterstein, Marc Schink, Thomas Schamberger, Lars Tebelmann, Manuel Ilg, Johann Heyszl,
2020-09-18



Secure IoT devices require side-channel protection

- “IoT Goes Nuclear” Ronen et al. [RSWO17]:
 - Attack on smart light bulb
 - Side-channel attack reveals firmware encryption key
 - Malicious firmware is deployed that spreads to other light bulbs
 - Firmware updates are crucial for long-running IoT devices
- Keys need to be protected against physical attacks

Can we retrofit side-channel protection to such devices?



Secure IoT devices require side-channel protection

- “IoT Goes Nuclear” Ronen et al. [RSWO17]:
 - Attack on smart light bulb
 - Side-channel attack reveals firmware encryption key
 - Malicious firmware is deployed that spreads to other light bulbs
 - Firmware updates are crucial for long-running IoT devices
- Keys need to be protected against physical attacks

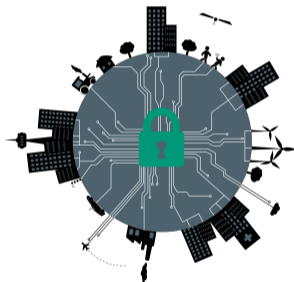
Can we retrofit side-channel protection to such devices?



Secure IoT devices require side-channel protection

- “IoT Goes Nuclear” Ronen et al. [RSWO17]:
 - Attack on smart light bulb
 - Side-channel attack reveals firmware encryption key
 - Malicious firmware is deployed that spreads to other light bulbs
- Firmware updates are crucial for long-running IoT devices
- Keys need to be protected against physical attacks

Can we retrofit side-channel protection to such devices?



Retrofitting side-channel protection to microcontrollers

■ We start with two observations:

1. Side-channel protections seems necessary but countermeasures like masking can be expensive
2. Microcontrollers often have crypto accelerators but without side-channel protection

Maybe we can use methods from leakage resilient cryptography

Retrofitting side-channel protection to microcontrollers

■ We start with two observations:

1. Side-channel protections seems necessary but countermeasures like masking can be expensive
2. Microcontrollers often have crypto accelerators but without side-channel protection

Maybe we can use methods from leakage resilient cryptography

Retrofitting side-channel protection to microcontrollers

■ We start with two observations:

1. Side-channel protections seems necessary but countermeasures like masking can be expensive
2. Microcontrollers often have crypto accelerators but without side-channel protection

Maybe we can use methods from leakage resilient cryptography

Retrofitting side-channel protection to microcontrollers

- We start with two observations:
 1. Side-channel protections seems necessary but countermeasures like masking can be expensive
 2. Microcontrollers often have crypto accelerators but without side-channel protection

Maybe we can use methods from leakage resilient cryptography

Retrofitting side-channel protection to microcontrollers

- We start with two observations:
 1. Side-channel protections seems necessary but countermeasures like masking can be expensive
 2. Microcontrollers often have crypto accelerators but without side-channel protection

Maybe we can use methods from leakage resilient cryptography

Leakage resilient cryptography

Overview

- **Algorithmic countermeasure** against side-channel attacks
- Bounds the leakage per execution such that an attacker cannot accumulate information
- Does not require high quality random numbers or specialized hardware

→ seems ideal for IoT devices where we have no control over the hardware

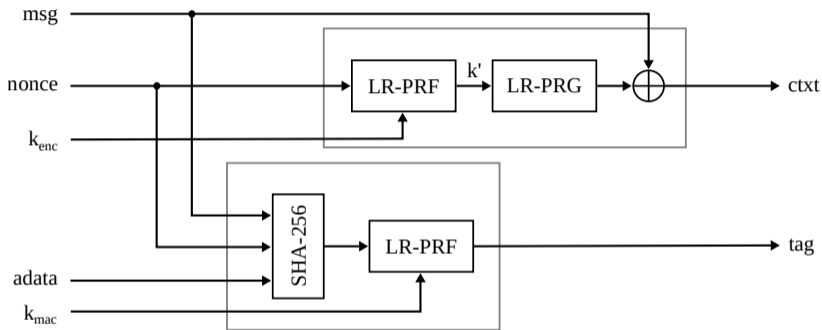
Leakage resilient cryptography

Overview

- **Algorithmic countermeasure** against side-channel attacks
 - Bounds the leakage per execution such that an attacker cannot accumulate information
 - Does not require high quality random numbers or specialized hardware
- **seems ideal for IoT devices where we have no control over the hardware**

Leakage resilient cryptography

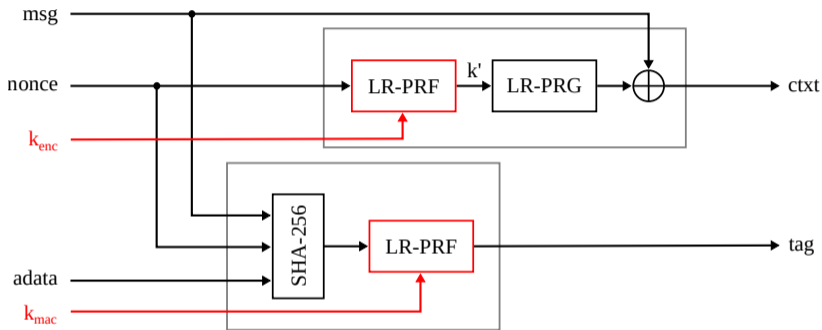
Leakage resilient authenticated encryption with associated data (LR-AEAD)



LR-AEAD proposed by Degabriele et al. [DJS19], security requirements relaxed by Krämer and Struck [KS20]

Leakage resilient cryptography

Leakage resilient authenticated encryption with associated data (LR-AEAD)



LR-AEAD proposed by Degabriele et al. [DJS19], security requirements relaxed by Krämer and Struck [KS20]

Leakage resilient cryptography

Leakage resilient pseudorandom function (LR-PRF), Medwed et al. [MSJ12]

- Implemented using standard block cipher, e.g., AES
- Side-channel resistance based on two methods:
 1. Limited data complexity, i.e. limited number of different operations under one key
 - Configuration parameter
 - Trade-off between security and performance
 2. Algorithmic noise from parallel S-boxes with equal inputs
 - Fully parallel implementations with 16 S-boxes are most effective
 - On microcontrollers we cannot influence the hardware implementations, typically 4 or 16 S-boxes in parallel

Leakage resilient cryptography

Leakage resilient pseudorandom function (LR-PRF), Medwed et al. [MSJ12]

- Implemented using standard block cipher, e.g., AES
- Side-channel resistance based on two methods:
 1. Limited data complexity, i.e. limited number of different operations under one key
 - Configuration parameter
 - Trade-off between security and performance
 2. Algorithmic noise from parallel S-boxes with equal inputs
 - Fully parallel implementations with 16 S-boxes are most effective
 - On microcontrollers we cannot influence the hardware implementations, typically 4 or 16 S-boxes in parallel

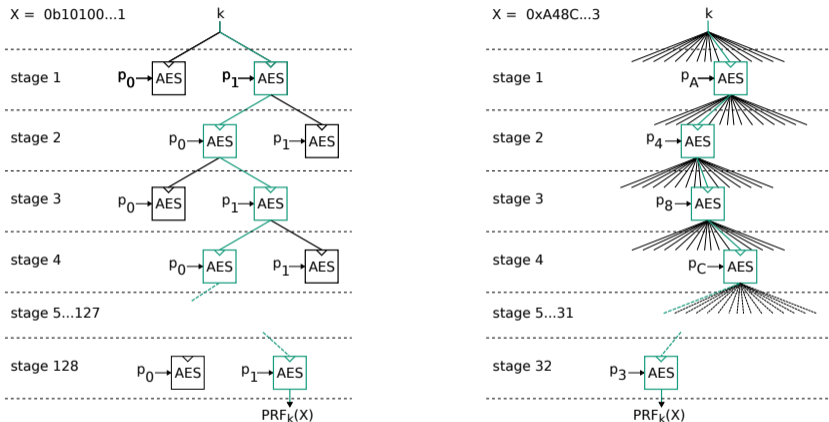
Leakage resilient cryptography

Leakage resilient pseudorandom function (LR-PRF), Medwed et al. [MSJ12]

- Implemented using standard block cipher, e.g., AES
- Side-channel resistance based on two methods:
 1. **Limited data complexity**, i.e. limited number of different operations under one key
 - Configuration parameter
 - Trade-off between security and performance
 2. **Algorithmic noise** from parallel S-boxes with equal inputs
 - Fully parallel implementations with 16 S-boxes are most effective
 - On microcontrollers we cannot influence the hardware implementations, typically 4 or 16 S-boxes in parallel

Leakage resilient cryptography

Leakage resilient pseudorandom function (LR-PRF), Medwed et al. [MSJ12]

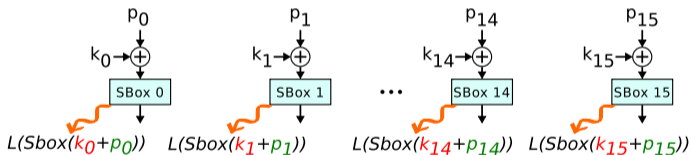


Dataflow graph of LR-PRF for data complexities 2 (left) and 16 (right)

Leakage resilient cryptography

Leakage resilient pseudorandom function (LR-PRF), Medwed et al. [MSJ12]

- Parallel S-Boxes add algorithmic noise
- Equal S-box inputs prevent divide-and-conquer:
 $p_0 = p_1 = \dots = p_{14} = p_{15}$, e.g. 0^{128} and 1^{128}



LR-PRF implementation on microcontrollers

Proposed construction

- The only hardware requirement for LR-PRF is an AES with parallel S-boxes
 - Often available on microcontrollers to enhance performance
- We can port this construction to microcontrollers:
 - We use existing hardware accelerator for the encryption in each LR-PRF stage
 - Protocol handling, key updates and generation of inputs is done in software
 - If we can secure this building block, we can implement the full LR-AEAD scheme



LR-PRF implementation on microcontrollers

Proposed construction

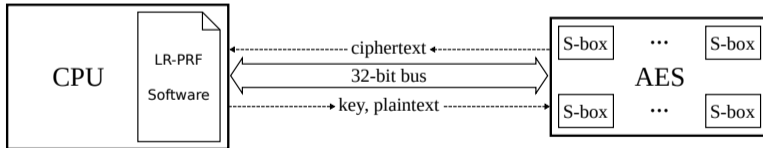
- The only hardware requirement for LR-PRF is an AES with parallel S-boxes
 - Often available on microcontrollers to enhance performance
- We can port this construction to microcontrollers:
 - We use existing hardware accelerator for the encryption in each LR-PRF stage
 - Protocol handling, key updates and generation of inputs is done in software
 - If we can secure this building block, we can implement the full LR-AEAD scheme



LR-PRF implementation on microcontrollers

Proposed construction

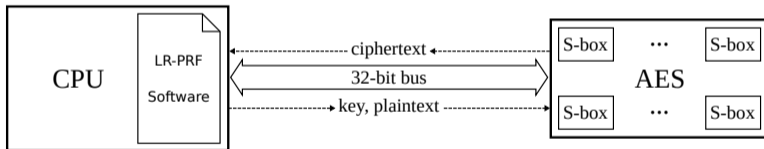
- The only hardware requirement for LR-PRF is an AES with parallel S-boxes
 - Often available on microcontrollers to enhance performance
- We can port this construction to microcontrollers:
 - We use existing hardware accelerator for the encryption in each LR-PRF stage
 - Protocol handling, key updates and generation of inputs is done in software
 - If we can secure this building block, we can implement the full LR-AEAD scheme



LR-PRF implementation on microcontrollers

Attack vectors

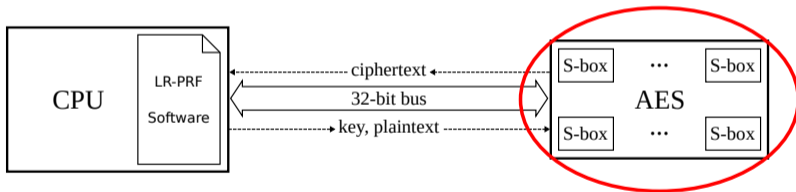
- We identify two attack vectors:
 1. Attacks on the AES accelerator
 2. Attacks on the (plain) key transfer



LR-PRF implementation on microcontrollers

Attack vectors

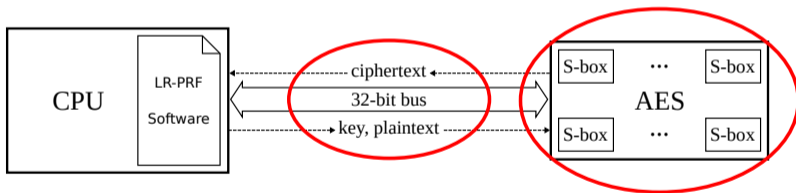
- We identify two attack vectors:
 1. Attacks on the AES accelerator
 2. Attacks on the (plain) key transfer



LR-PRF implementation on microcontrollers

Attack vectors

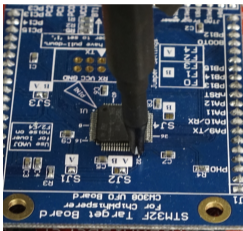
- We identify two attack vectors:
 1. Attacks on the AES accelerator
 2. Attacks on the (plain) key transfer



Side-channel evaluation

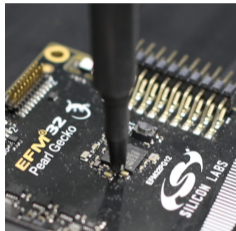
Devices under test

STM32



- STMicroelectronics STM32F215RE (90nm)
- ARM Cortex-M3
- AES with parallel 16 S-boxes
- SHA-256 implemented in software

EFM32



- Silicon Labs EFM32PG12 (90nm)
- ARM Cortex-M4
- AES with 4 parallel S-boxes
- SHA-256 accelerator

Side-channel evaluation

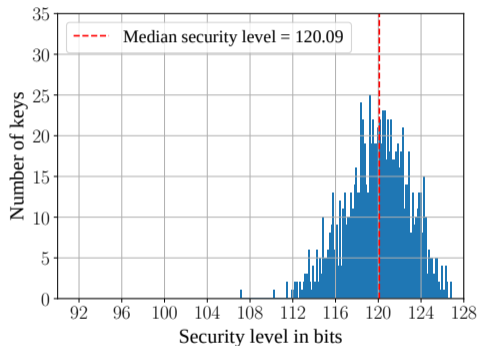
Setup

- For low data complexities, attack success rate depends on value of the key
- We attack multiple keys and give statistics over the security level
- Evaluation setup:
 - EM probe with 2.5mm diameter coil
 - Correlation based leakage tests to detect points of interest
 - Multivariate template attacks on key bytes

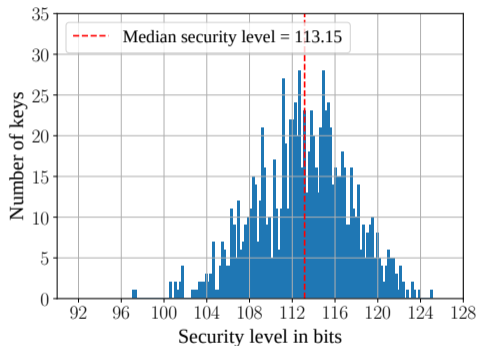
Side-channel evaluation

Attack on key transfer between CPU and accelerator

STM32 - 16 S-boxes



EFM32 - 4 S-boxes

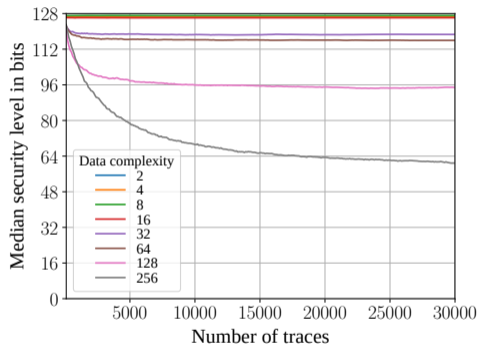


Security level for 1,000 random keys after template attack on the key transfer

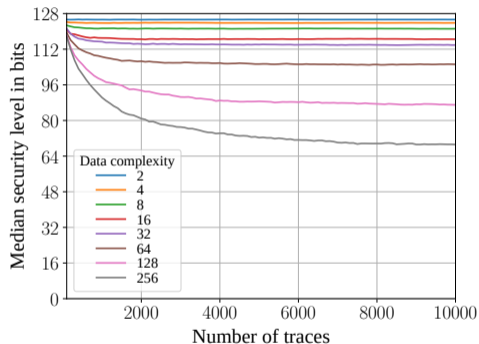
Side-channel evaluation

Attack on AES encryption by the accelerator

STM32 - 16 S-boxes



EFM32 - 4 S-boxes

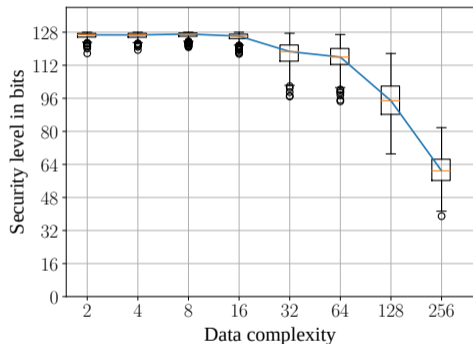


Median security levels from 300 random keys after template attack on the AES

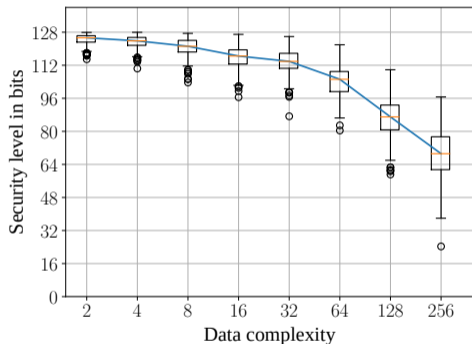
Side-channel evaluation

Attack on AES encryption by the accelerator

STM32 - 16 S-boxes



EFM32 - 4 S-boxes



Distribution of security levels of 300 random keys after template attack on the AES using the maximum amount of traces

Our construction provides resilience against SCA attacks in an IoT scenario

1. For both controllers we find implementations with security levels above 100 bits
 - Efficient protection using leakage resilience is achievable on commodity devices
 - Enables, e.g., secured firmware updates
 2. Small memory overhead and performance impact:
 - Requires only about 1 percent of available memory
 - Impact of LR-PRF amortizes over length of input data, full analysis in paper
 3. Applicable to a wide range of microcontrollers
 - Only requires AES accelerators with parallel S-boxes
 - Hardware hash accelerators can be used if available
- Source code for full LR-AEAD available at
<https://github.com/fraunhofer-aisec/leakres-aead-microcontroller>

Our construction provides resilience against SCA attacks in an IoT scenario

1. For both controllers we find implementations with security levels above 100 bits
 - Efficient protection using leakage resilience is achievable on commodity devices
 - Enables, e.g., secured firmware updates
2. Small memory overhead and performance impact:
 - Requires only about 1 percent of available memory
 - Impact of LR-PRF amortizes over length of input data, full analysis in paper
3. Applicable to a wide range of microcontrollers
 - Only requires AES accelerators with parallel S-boxes
 - Hardware hash accelerators can be used if available

■ Source code for full LR-AEAD available at
<https://github.com/fraunhofer-aisec/leakres-aead-microcontroller>

Our construction provides resilience against SCA attacks in an IoT scenario





1. For both controllers we find implementations with security levels above 100 bits
 - Efficient protection using leakage resilience is achievable on commodity devices
 - Enables, e.g., secured firmware updates
2. Small memory overhead and performance impact:
 - Requires only about 1 percent of available memory
 - Impact of LR-PRF amortizes over length of input data, full analysis in paper
3. Applicable to a wide range of microcontrollers
 - Only requires AES accelerators with parallel S-boxes
 - Hardware hash accelerators can be used if available

■ Source code for full LR-AEAD available at
<https://github.com/fraunhofer-aisec/leakres-aead-microcontroller>

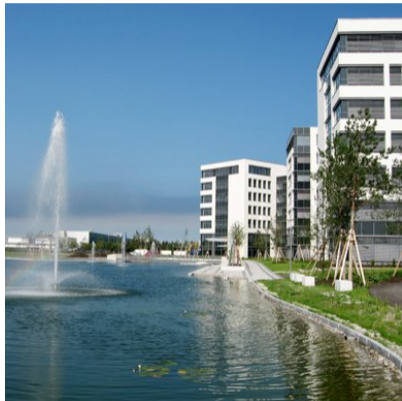
Our construction provides resilience against SCA attacks in an IoT scenario

1. For both controllers we find implementations with security levels above 100 bits
 - Efficient protection using leakage resilience is achievable on commodity devices
 - Enables, e.g., secured firmware updates
 2. Small memory overhead and performance impact:
 - Requires only about 1 percent of available memory
 - Impact of LR-PRF amortizes over length of input data, full analysis in paper
 3. Applicable to a wide range of microcontrollers
 - Only requires AES accelerators with parallel S-boxes
 - Hardware hash accelerators can be used if available
- Source code for full LR-AEAD available at
<https://github.com/fraunhofer-aisec/leakres-aead-microcontroller>

References

-  Jean Paul Degabriele, Christian Janson, and Patrick Struck, *Sponges resist leakage: The case of authenticated encryption*, ASIACRYPT, Lecture Notes in Computer Science, vol. 11922, Springer, 2019, pp. 209–240.
-  Juliane Krämer and Patrick Struck, *Leakage-resilient authenticated encryption from leakage-resilient pseudorandom functions*, Cryptology ePrint Archive, Report 2020/280, 2020.
-  Marcel Medwed, François-Xavier Standaert, and Antoine Joux, *Towards super-exponential side-channel security with efficient leakage-resilient prfs*, CHES, Lecture Notes in Computer Science, vol. 7428, Springer, 2012, pp. 193–212.
-  Eyal Ronen, Adi Shamir, Achi-Or Weingarten, and Colin O’Flynn, *IoT goes nuclear: Creating a ZigBee chain reaction*, IEEE Symposium on Security and Privacy, IEEE Computer Society, 2017, pp. 195–212.

Contact information



Florian Unterstein

Infineon Technologies AG*
florian.unterstein@infineon.com

Marc Schink, Manuel Ilg, Johann Heyszl

Fraunhofer Institute for Applied and Integrated Security AISEC
<firstname.lastname>@aisec.fraunhofer.de

Thomas Schamberger, Lars Tebelmann

Technical University of Munich
t.schamberger@tum.de, lars.tebelmann@tum.de

*work was done while with Fraunhofer Institute for Applied and Integrated Security AISEC