Fuzzy Extractors using Low-Density Parity-Check Codes

Nico Mexis[1], Nikolaos Athanasios Anagnostopoulos[1,2], Tolga Arul[1,2], Florian Frank[1], and Stefan Katzenbeisser[1]

[1]Faculty of Computer Science and Mathematics, University of Passau, Germany
[2]Department of Computer Science, Technical University of Darmstadt, Germany

33rd Crypto Day, 17. September 2021

As biometric data become more and more important for authentication, adequate mechanisms are required in order to filter out noise or general errors in the acquirement of these data. These days, the notion of biometric data has been extended from humans to digital devices. Very popular examples of such device fingerprints are secure device IDs (e.g., the Secure Device ID described in the 802.1ar standard proposed by IEEE (2018)), unique keys generated from Trusted Platform Modules (TPMs), or unique responses and keys produced from Physical Unclonable Functions (PUFs). However, just like human fingerprints can be disrupted by sweat, scratches, poor image resolution, or other similar issues, these device authentication methods suffer from noise, or their dependencies to ambient temperature and other environmental factors.

To circumvent these problems, fuzzy extractors may be deployed to correct minor errors in the identification/authentication data collected. A fuzzy extractor takes a first collection of data w and generates helper data h and a key R. This key R can now be used for authentication in the following way. If new data w' are collected, the fuzzy extractor will try to reconstruct R out of w' using the previously generated helper data h. It is important to note that most attacks are prevented due to the fact that h should not yield enough information about R to allow an attacker to gain an advantage for the reconstruction of R. Most often, reverse fuzzy extractor schemes are used in order to error correct identification/authentication protocols in practice. Such schemes, which are examined in detail in the work of Van Herrewege, Katzenbeisser, Maes, Peeters, Sadeghi, Verbauwhede & Wachsmann (2012), separate the fuzzy extractor scheme so that the more computationally intensive procedures are executed on the more capable verification device.

In order to generate the needed helper data, an error correction code is being utilised in the context of a fuzzy extractor. Most works propose the usage of Bose–Chaudhuri–Hocquenghem (BCH) codes due to their performance and cyclicity. We chose to use Low-Density Parity-Check (LDPC) codes instead, due to their linearity and their utilisation of sparse parity-check matrices. Since their discovery in the 1960s, many new algorithms and procedures have been proposed that allow for a more efficient implementation of LDPC codes, as they are able to encode and decode messages faster and store the underlying parity-check matrices more efficiently. Contrary to other error correction codes, LDPC codes offer, depending on the decoding algorithm being utilized, fast and optimised procedures for generating helper data. Since it has been proven by Sae-Young Chung, Forney, Richardson & Urbanke (2001) and Newagy, Fahmy & El-Soudani (2007) that LDPC codes can reach code rates near the Shannon limit, only very limited helper data have to be produced. Many of the relevant parity check matrices are already known to be very practical.

We have implemented in Java a fuzzy extractor utilising an LDPC code. Our implementation employs the Sum-Product algorithm (Belief Propagation) and is based upon the fuzzy extractor scheme examined in the work of Kang, Hori, Katashita & Hagiwara (2013). The relevant source code has been published under the MIT License[1,2]. We have also tested the practicability of these fuzzy extractors by deploying them into an existing PUF-based security solution for a proof-of-concept System of Systems (SoS) implementation, which was previously introduced by Mexis, Anagnostopoulos, Chen, Bambach, Arul & Katzenbeisser (2021). The performance of the system

[1]`https://github.com/ThexXTURBOXx/LDPC`
[2]`https://github.com/ThexXTURBOXx/FuzzyExtractors`

crypto day matters 33 (2021)
Beiträge vom 33. Kryptotag, 17.09.2021, online.
doi:10.18420/cdm-2021-33-41

is examined by measuring the time needed for the code to generate helper data out of a given PUF response for a particular challenge (Figure 1 (a)), the time needed to error correct another response produced by the same PUF device for the same challenge and reconstruct the legitimate key (Figure 1 (b)), and the time needed for our implementation to identify and reject a PUF response stemming from a different PUF device, as it cannot error correct it in a way that reproduces the legitimate key (Figure 1 (c)).

As demonstrated in Figure 1, our implementation offers fast and simple access to fuzzy extraction capability using LDPC codes. Since all of the performance times measured on a typical Internet-of-Things (IoT) device, namely the Raspberry Pi 3B+, are less than 600 ms, our LDPC-based fuzzy extractor implementation allows for the realisation of practical identification and authentication schemes.

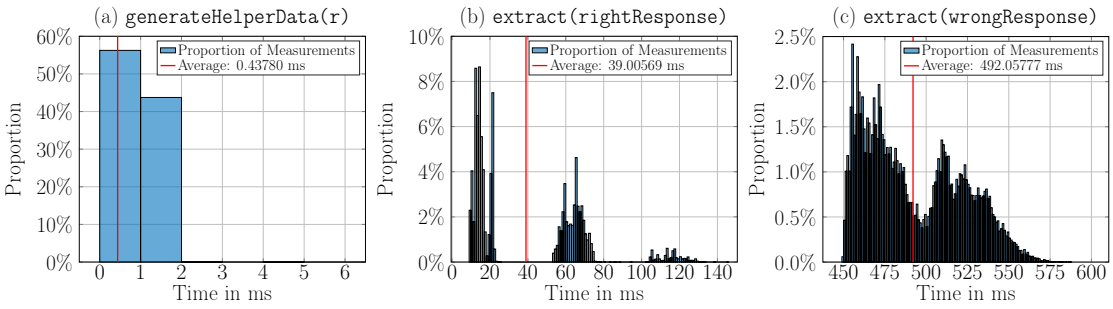

Figure 1: Measuring the performance of our LDPC-based fuzzy extractor implementation.

References

IEEE (2018). Standard for Local and Metropolitan Area Networks - Secure Device Identity – IEEE Standard 802.1AR-2018 (Revision of IEEE Standard 802.1AR-2009). URL `https://doi.org/10.1109/ieeestd.2018.8423794`.

Hyunho Kang, Y. Hori, T. Katashita & M. Hagiwara (2013). The Implementation of Fuzzy Extractor is Not Hard to Do: An Approach Using PUF Data. In *Proceedings of the 30th Symposium on Cryptography and Information Security, Kyoto, Japan*, 22–25. URL `https://www.jst.go.jp/crest/dvlsi/list/SCIS2013/pdf/SCIS2013_2E1-5.pdf`.

Nico Mexis, Nikolaos Athanasios Anagnostopoulos, Shuai Chen, Jan Bambach, Tolga Arul & Stefan Katzenbeisser (2021). A Lightweight Architecture for Hardware-Based Security in the Emerging Era of Systems of Systems. *ACM Journal on Emerging Technologies in Computing Systems* **17**(3). ISSN 1550-4832. URL `https://doi.org/10.1145/3458824`.

Fatma A. Newagy, Yasmine A. Fahmy & Magdi M. S. El-Soudani (2007). Designing Near Shannon Limit LDPC Codes Using Particle Swarm Optimization Algorithm. In *Proceedings of the 2007 IEEE International Conference on Telecommunications and the Malaysia International Conference on Communications*, 119–123. IEEE. URL `https://ieeexplore.ieee.org/document/4448612/`.

Sae-Young Chung, G.D. Forney, T.J. Richardson & R. Urbanke (2001). On the Design of Low-Density Parity-Check Codes within 0.0045 dB of the Shannon Limit. *IEEE Communications Letters* **5**(2), 58–60. ISSN 1089-7798. URL `http://ieeexplore.ieee.org/document/905935/`.

Anthony Van Herrewege, Stefan Katzenbeisser, Roel Maes, Roel Peeters, Ahmad-Reza Sadeghi, Ingrid Verbauwhede & Christian Wachsmann (2012). Reverse Fuzzy Extractors: Enabling Lightweight Mutual Authentication for PUF-Enabled RFIDs. In *Financial Cryptography and Data Security – 16th International Conference, FC 2012*, volume 7397 of *Lecture Notes in Computer Science (LNCS)*, 374–389. Springer, Berlin & Heidelberg. ISBN 978-3-642-32946-3 / 978-3-642-32945-6. URL `https://doi.org/10.1007/978-3-642-32946-3_27`.