Fuzzy Extractor Documentation

Release 0.2

Carter Yagemann

Mar 21, 2018

Contents:

CHAPTER 1

Introduction

Fuzzy extractors are a cryptography primitive designed to reliably derive keys from noisy sources. This makes them suitable for areas like biometric authentication where two measurements of the same subject can yield slightly different values. This implementation uses hamming distance as its error metric, meaning that two binary strings will produce the same key with very high probability if their hamming distance is within some given threshold.

The storage and retrieval of keys is performed using a primitive known as a *digital locker*. More information is available in the references section of this documentation.

Note that this is a probabilistic primitive based on very recent research. Use this library in real security applications at your own risk, ideally after performing some empirical evaluation for your chosen thresholds.

CHAPTER 2

Installing

This library can be install from pip:

```
$ pip install fuzzy-extractor
```

2.1 Development

This repository comes with a *Makefile* to help with getting a development environment configured:

```
$ make help
```

CHAPTER 3

Usage

3.1 Getting Started

This section will cover the basics of using fuzzy extractors. First, we need to create an extractor:

```
from fuzzy_extractor import FuzzyExtractor

extractor = FuzzyExtractor(16, 8)
```

The extractor we just created will accept 16 byte (128-bit) input values and guarantees that inputs within 8 bits of each other will produce the same key with over 0.9999 probability (see the references for more details).

We're now ready to generate a key for some input:

```
key, helper = extractor.generate('AABBCCDDEEFFGGHH')
```

Note that *generate()* returned two things: *key* and *helper*. The former is the secret that can now be used for further cryptography. The latter does not need to be protected (i.e., it is not a secret), but it does need to be stored somewhere if we want to be able to reproduce the same key later.

As long as we have the public helper, we can reproduce the key with any input close enough to the original:

```
r_key = extractor.reproduce('AABBCCDDEEFFGGHH', helper)  # r_key should equal key
r_key = extractor.reproduce('AABBCCDDEEFFGGHI', helper)  # r_key will probably still␣
↪equal key!
r_key = extractor.reproduce('AAAAAAAAAAAAAAAA', helper)  # r_key is no longer likely␣
↪to equal key
```

CHAPTER 4

FuzzyExtractor

A Python implementation of fuzzy extractor

class fuzzy_extractor.**FuzzyExtractor**(*length*, *ham_err*, *rep_err=0.001*, ***locker_args*)

The most basic non-interactive fuzzy extractor

__init__(*length*, *ham_err*, *rep_err=0.001*, ***locker_args*)

Initializes a fuzzy extractor

Parameters

- **length** – The length in bytes of source values and keys.
- **ham_err** – Hamming error. The number of bits that can be flipped in the source value and still produce the same key with probability (1 - rep_err).
- **rep_err** – Reproduce error. The probability that a source value within ham_err will not produce the same key (default: 0.001).
- **locker_args** – Keyword arguments to pass to the underlying digital lockers. See parse_locker_args() for more details.

parse_locker_args(*hash_func='sha256'*, *sec_len=2*, *nonce_len=16*)

Parse arguments for digital lockers

Parameters

- **hash_func** – The hash function to use for the digital locker (default: sha256).
- **sec_len** – security parameter. This is used to determine if the locker is unlocked successfully with accuracy (1 - 2 ^ -sec_len).
- **nonce_len** – Length in bytes of nonce (salt) used in digital locker (default: 16).

generate(*value*)

Takes a source value and produces a key and public helper

This method should be used once at enrollment.

Note that the "public helper" is actually a tuple. This whole tuple should be passed as the helpers argument to reproduce().

Parameters **value** – the value to generate a key and public helper for.

Return type (key, helper)

reproduce(*value*, *helpers*)

Takes a source value and a public helper and produces a key

Given a helper value that matches and a source value that is close to those produced by generate, the same key will be produced.

Parameters

- **value** – the value to reproduce a key for.
- **helpers** – the previously generated public helper.

Return type key or None

CHAPTER 5

References

- Canetti, Ran, et al. “Reusable fuzzy extractors for low-entropy distributions.” *Annual International Conference on the Theory and Applications of Cryptographic Techniques*. Springer, Berlin, Heidelberg, 2016.

CHAPTER 6

License

CHAPTER 7

Indices and tables

- genindex
- modindex
- search

Python Module Index

f

Index

Symbols

F

G

P

R