shuffled Documentation

Release dev

Bertrand Bonnefoy-Claudet

October 09, 2016

Contents

CHAPTER 1

How to Use

class shuffled.**Shuffled**(*range_size*, *seed=None*)

Randomized integer ranges

Parameters

- **range_size** (*int*) – Size of the range
- **seed** (*bytes*) – Seed to make randomization repeatable

```
>>> shuffled_range = Shuffled(10)
>>> list(shuffled_range)
[4, 1, 2, 9, 8, 5, 3, 0, 6, 7]
>>> same_shuffled_range = Shuffled(10, seed=shuffled_range.seed)
>>> list(same_shuffled_range)
[4, 1, 2, 9, 8, 5, 3, 0, 6, 7]
```

seed

Seed of the randomization.

It can be used to create a new identical *Shuffled* object.

CHAPTER 2

Internal APIs

Note: Use the following APIs at your own risk.

2.1 Index Randomization

class shuffled.crypto.**AesRandomizer**(*key*)

domain_size = 340282366920938463463374607431768211456

randomize(*integer*)

class shuffled.crypto.**IndexEncryptor**(*randomizers*, *size*)
Encrypt indexes using pseudo-random function.

Parameters

- **randomizers** – List of instances with an appropriate pseudo-random `randomize` method and `domain_size` integer attribute, such as *AesRandomizer* objects.
- **size** (*int*) – Size of the domain

encrypt(*index*)
Permutation of `range(self.size)`

Parameters **index** (*int*) – Integer in `range(self.size)`

2.2 Generalized-Feistel Cipher

shuffled.feistel.**encrypt**(*round_functions*, *a*, *b*, *m*, *size*)
Generalized-Feistel encryption

Parameters

- **round_functions** (*List[int -> int]*) – List of pseudo-random functions with values in `range(n)` where `n >= size`
- **a** (*int*) – Positive integer
- **b** (*int*) – Positive integer

- **m** (*int*) – Message to encrypt in `range(size)`
- **size** (*int*) – Size of the domain

The algorithm comes from Black and Rogaway (Ciphers with Arbitrary Finite Domains, 2002).

CHAPTER 3

Indices and tables

- genindex
- modindex
- search

S

A

D

E

I

R

S