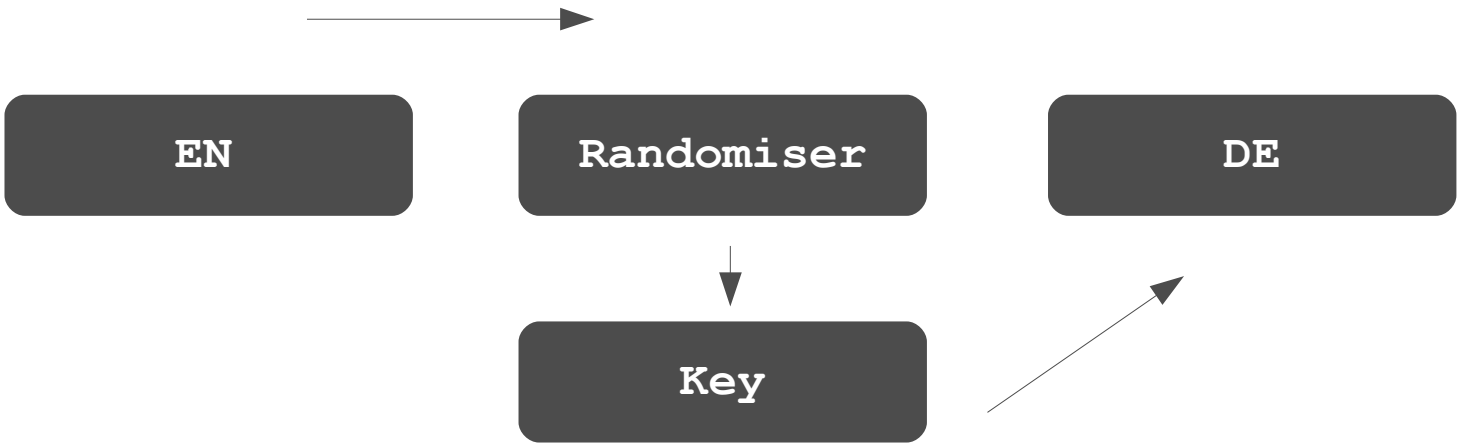
Simple XOR-Like

One-Time-Pad Encryption/Decryption Software

By
A'mmer Almadani

Layout of Procedure

Encryption/General Case(Messages)



EN= Source File(message, code, folder)
DE= Encrypted File(Output)

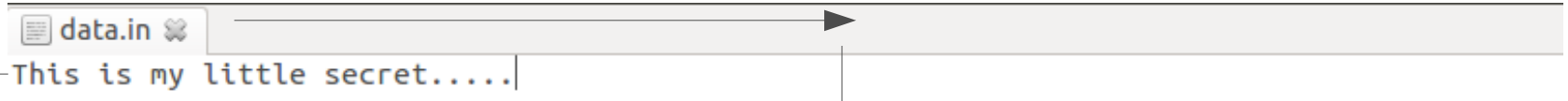
The Layout above could be written as follows:
EN (+) Key = DE (+) Key

Definition

- EN or en:
  - The input string; binary in case of compressed files and plain text in case of messages.
- DE or de:
  - The output string; the encrypted data
- Randomiser or rand:
  - The piece of code that insures every attempt differs from the next.
- Key or key:
  - Between en and de, the string generated from the randomiser will produce the key.

Example

Encrypting a message



This is my little secret.....

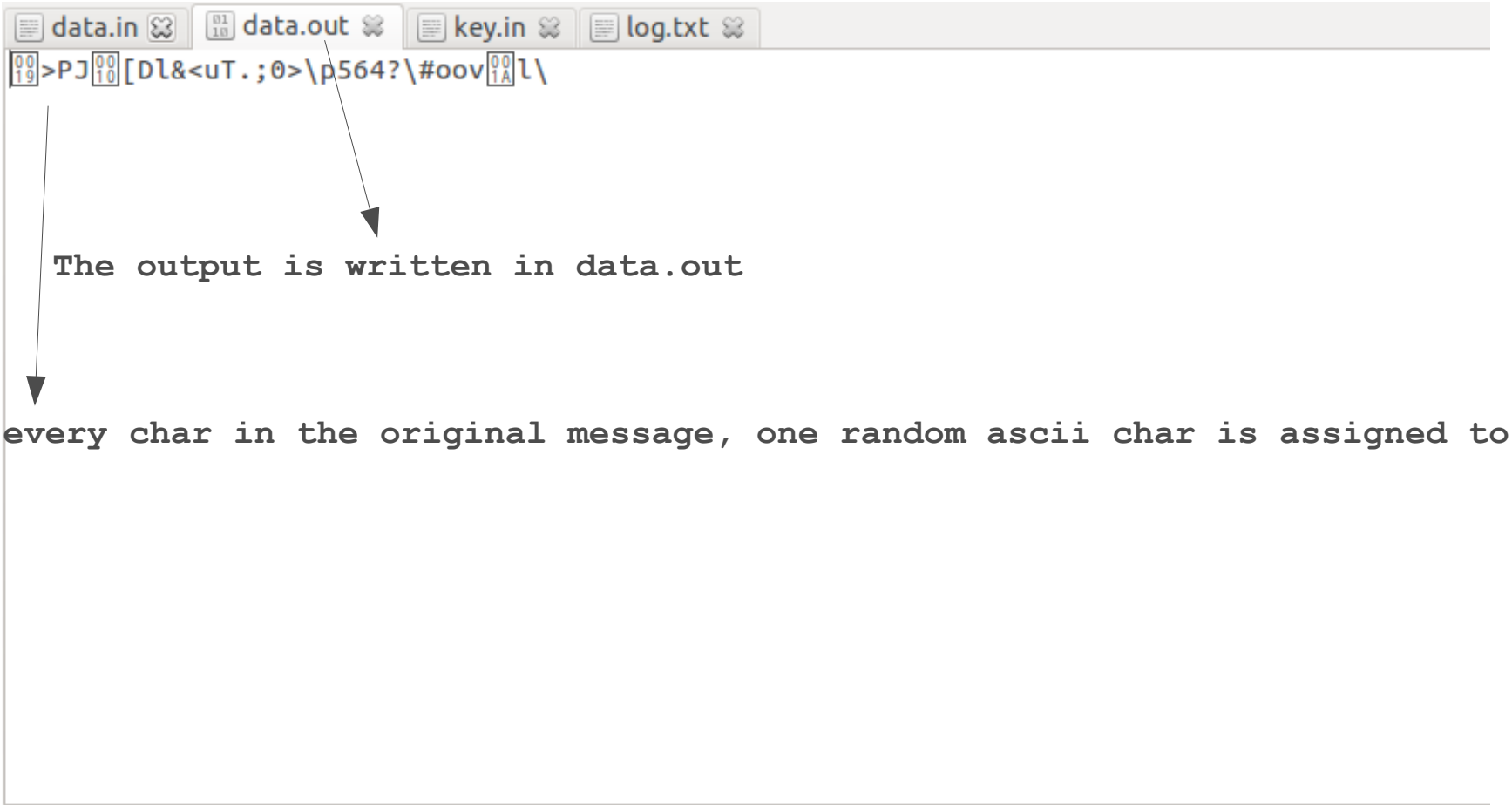
The source file(in this case a message) is written in data.in

The message is only limited by your device's computation abilities

NOTE: The old version used Gedit as its default text editor; the new one is nano

Example

Encrypting a message



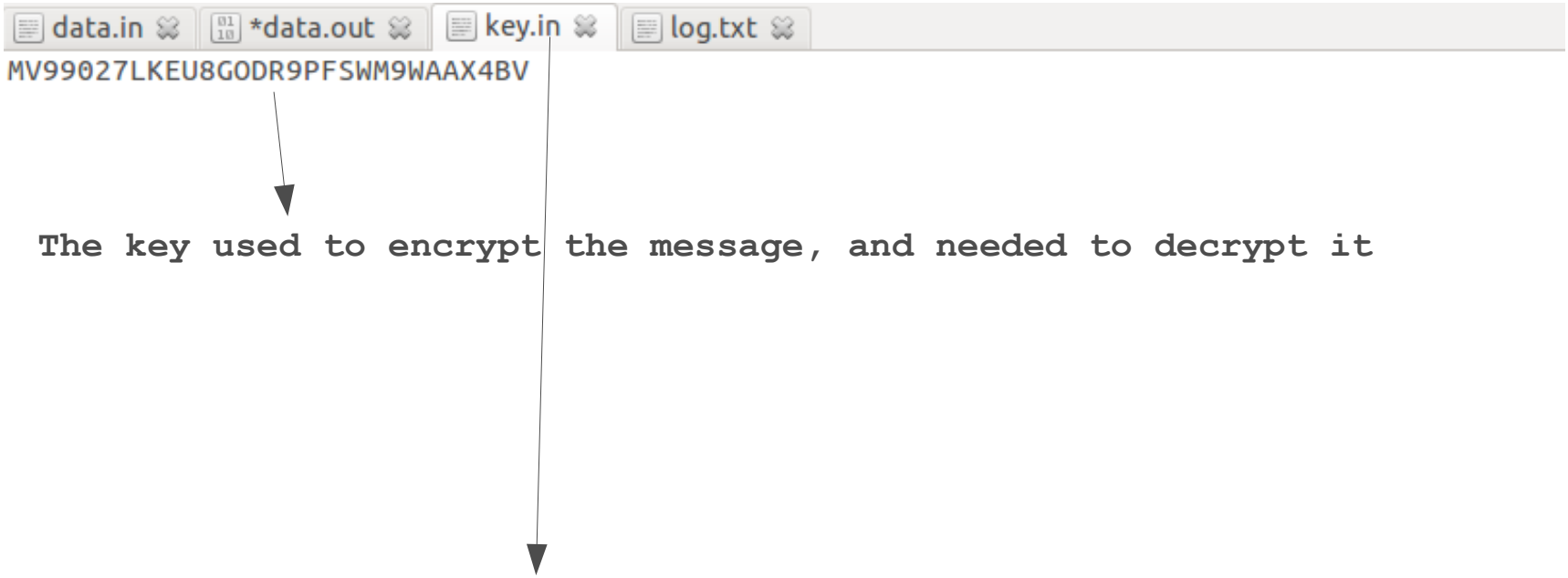
The output is written in data.out

For every char in the original message, one random ascii char is assigned to it

NOTE: The old version used Gedit as its default text editor; the new one is nano

Example

Encrypting a message



The key used to encrypt the message, and needed to decrypt it

The key is written in key.in

NOTE: The old version used Gedit as its default text editor; the new one is Nano.

The old version used uppercase and digits to encrypet,
The newer uses uppercase and lowercase

Example

Encrypting a message

data.in | *data.out | key.in | log.txt

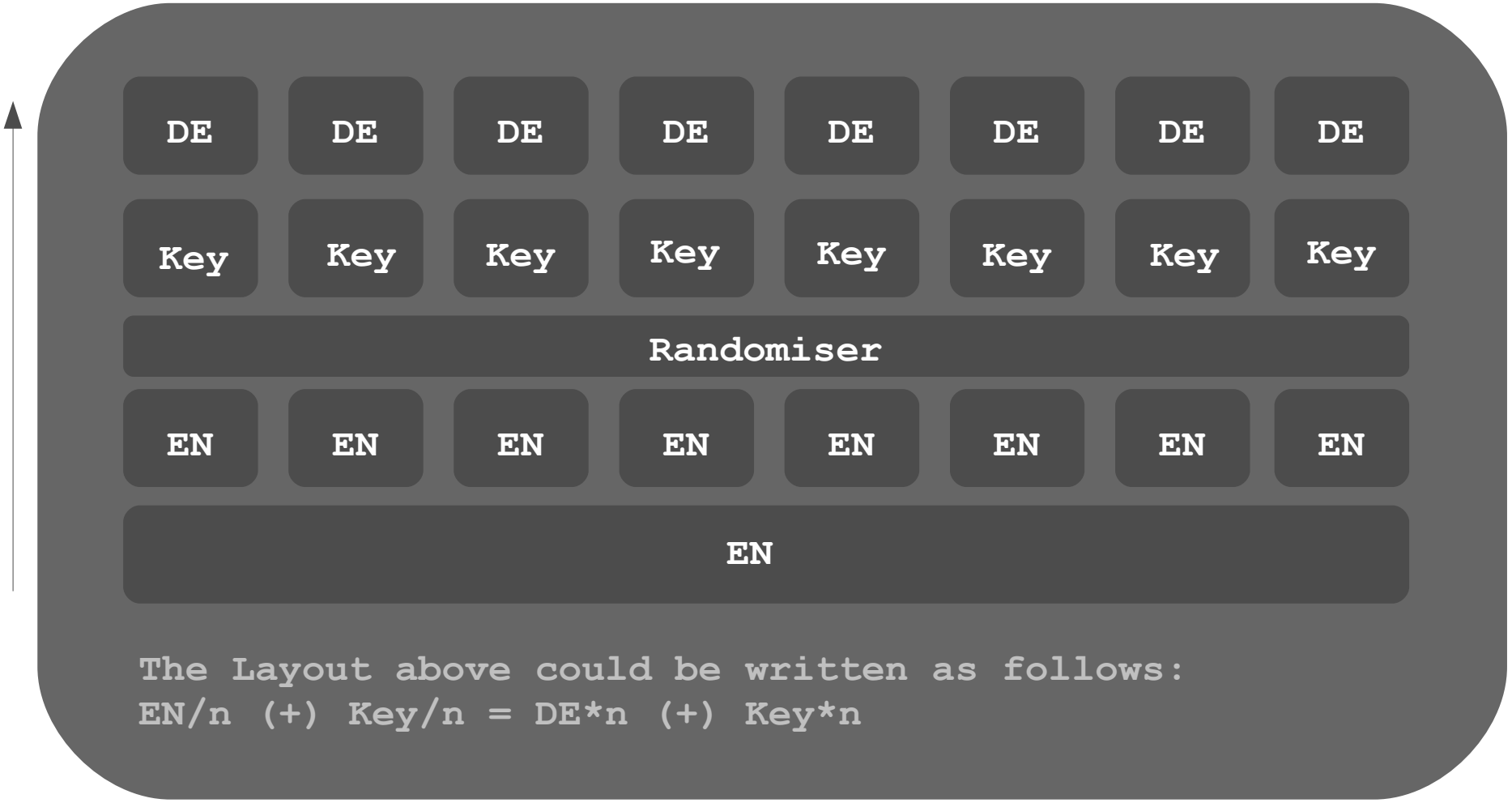
```
Date: 2012-09-05 15:20:19.934346
Original Message: This is my little secret.....

Number of Characters: 30
Key:MV99027LKEU8GODR9PFSWM9WAAX4BV
Encrypted Message: >PJ[Dl&<uT.;0>\p564?\#oov l\
```

All data are logged in log.txt

NOTE: The old version used Gedit as its default text editor; the new one is nano

Layout of Procedure

Encryption/Real-Time Communication(Messaging Client)



Note: not implemented yet

Issues

Encryption/Real-Time Communication(Messaging Client)

- Visibility of the key over the internet
  - Because Simple XOR does not meet the most important rule; the key must never be accessible to anyone except the recipient, dividing the message into increments then running the randomiser was necessary.
  - With the latter in motion, the visibility of the key/keys will hinder any exploitive attempts in a real-time communication environment. Every message is divided, and every quotient has a key.

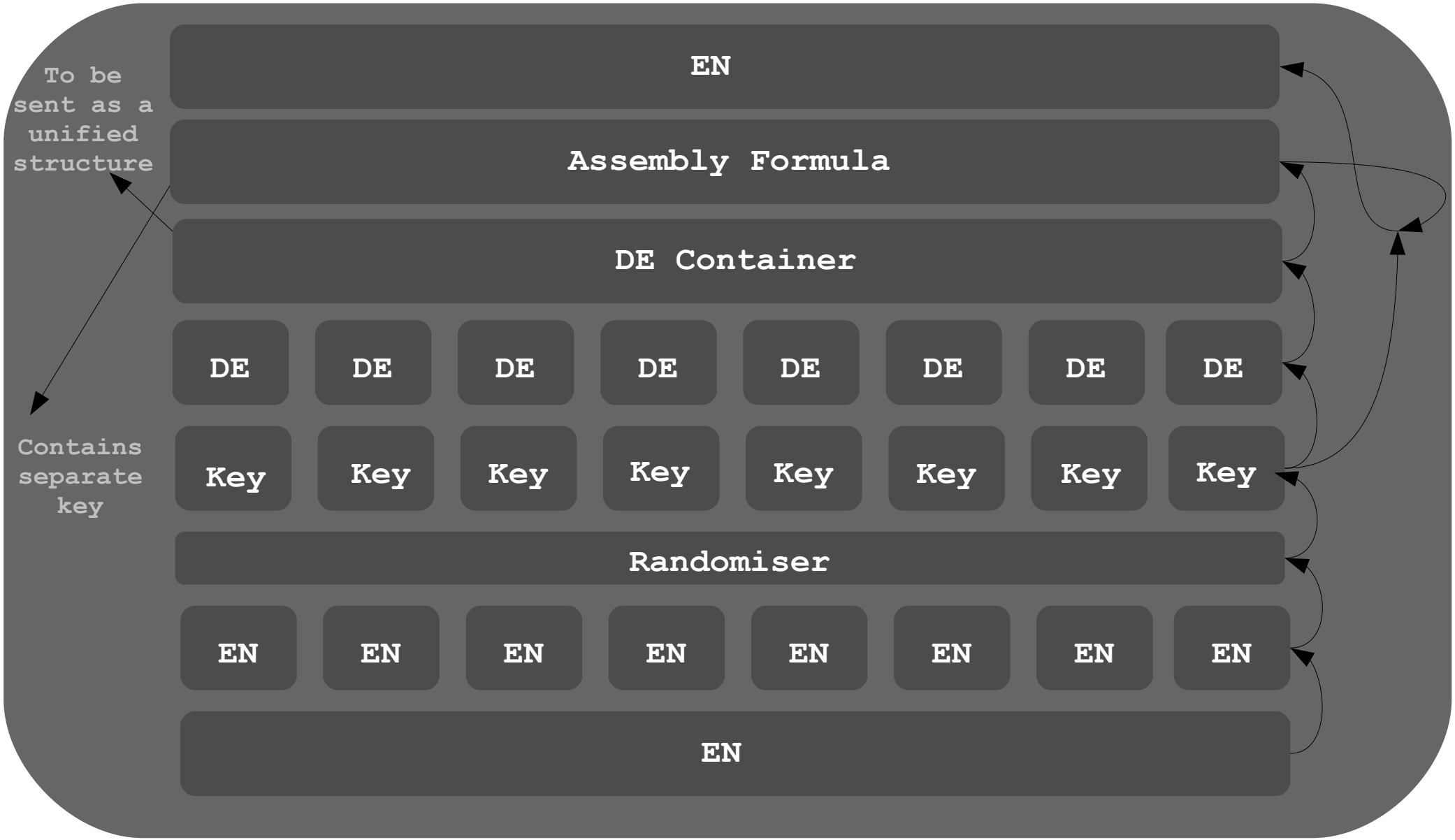
Issues

Encryption/Real-Time Communication(Messaging Client)

- Data corruption
  - It is very likely that decrypting the files(divided) will output a corrupted file. Because of this, the encrypted files(divided) must be contained
- Division
  - Using a visible mathematical equation to divide “en” and “key” will defy the purpose of the encryption

Theory

Encrypt&Decrypt/Real-Time Communication(Messaging Client)



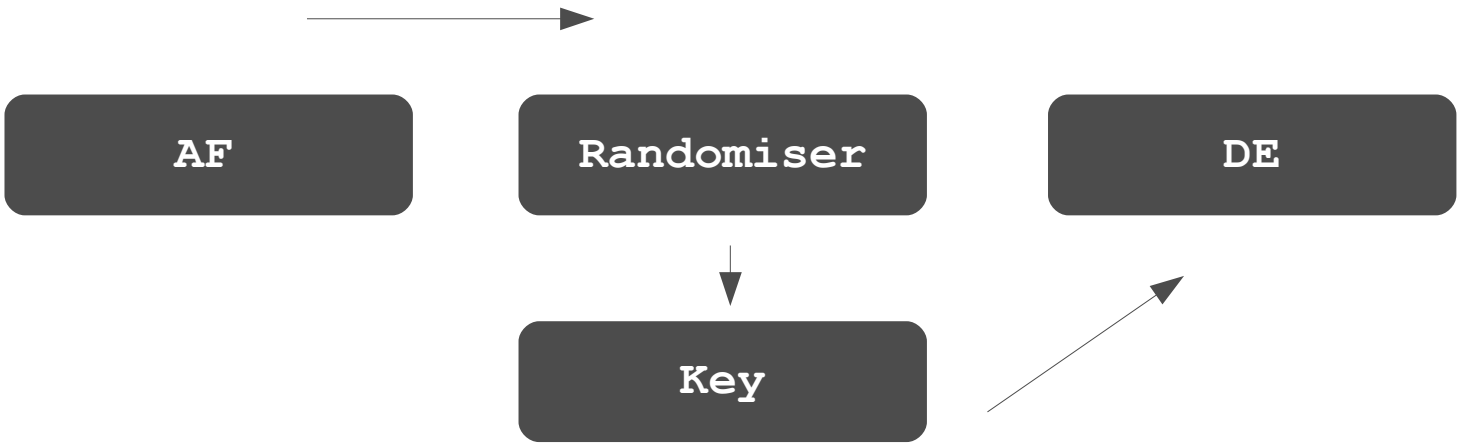
Note: not implemented yet

Definition

- DE Container
  - The container is needed to send the encrypted source as a unified structure; the data inside is still divided.
- AF or Assembly Formula
  - A user defined formula; this is sent to the recipient using the standard encryption. It has its own separate key.

Assembly Formula

Encrypt&Decrypt/Real-Time Communication(Messaging Client)



The Assembly Formula is a user defined mathematical equation.

Note: not implemented yet

DE Container

Encrypt&Decrypt/Real-Time Communication(Messaging Client)

Note: not implemented yet

To ease the process of data delivery and avoid data corruption