CODING
Programs for Coding “in Place”

Project CODING

by

Dipl.-Math. Jürgen Müller

D-64289 Darmstadt

Level: 29/11/2013

Motivation

"Aha, still another symmetric coding system, how boring !"
That isn't quite correct !

- Both encoding and decoding reaches data throughput **results of 2-digit megabytes per second**.
- There are **no trapdoor functions**, i.e. the CODING-algorithm is absolutely "clean" converted. In case of doubt you can download the source code and construct your own executable files using the Free Pascal compiler (http://www.freepascal.org/) and the Lazarus development system (http://sourceforge.net/projects/lazarus/). Nobody neither the author nor other powerful organizations has the possibility in any way to deduce a non trivial key without checking all possible keys.
- With CODING you can use a **special method to enter a key**, the so-called **"overlaying"**. You know that: You are advised to use a key as complex as possible and you also have to change the key every 3 months please ! But why you have to adapt the technique ? Why the technique cannot adapt to you, why you don't have the possibility to use handy and therefore easy to remember keys ?

  You are completely right ! CODING handles that for you !

  How to do it ? Very easy:
  You do not enter only an ordinary key but you also enter a name or a set of names of datasets, which content is used to "modify" the key.
  But hasn't a "rogue" the possibility to reproduce this and to consider this by trying to decode ?
  Yes and no ! Up to all information of any overlaying dataset can be used to modify the initial key. Doing so not all information of a overlaying dataset is selected and used. Rather this will be controlled by additional parameters and the respective key values at the referring positions. Because of that the "rogue" has **no other possibility** than **to overlay every possible key** used to decode **by overlay dataset information** once again, **even if he/she knows the overlay datasets and the additional parameters !**
  This action to detect an even simple key reduces the number of decoding tests per time unit in a deciding way, so **the exposing effort of your simple key in conjunction with overlaying datasets beats the exposing effort of a complex key without overlaying dataset by far !**
  If however the overlaying datasets and the additional parameters are unknown, your real key is generated by your "simple" key duplicated to fill the maximum key length and then this intermediate key is "coincidentally" overwritten at single or multiple positions by overlaying dataset information. **Therefore your real key is an unsystematic key of maximum key length, so the "rogue" has to prove all such keys !**

  As you see, the remembering of complex passwords (which you can't remember, therefore you write them down anyway, what's wrong on the other hand) is brought to an end. Don't remember complex rules, the right place for your password is your brain so you can remember it in a simple way !

Outline

1 Description of the Method

The used method of the programs CODINGx (x=1,2,3,...) is suitable for ciphering any data in place. The method is characterised by

- the feature that input data and output data (coded data) are of the **same length**,
- a **high operation velocity** (e.g. 2-digit MB value / sec. having a block length of 512 bytes, dimension t=1 [see 1.4 above], PC of the author),
- **data blocks**, **independently** processed, **of** (nearly) **any block length** (e.g. if a block can't be deciphered, all other blocks can; the block length can vary inside one dataset),
- the possibility of **stream ciphering** in form of block cipher as well as **using variable block length** and **extensive protection against compromising**,
- a **very high safety** growing in any region by appropriately dimensioning (dimension t) the method (**à** http://eprint.iacr.org/2013/742.pdf),
- the possibility to generate a "wide range" of **authentication and hash methods** in (nearly) **any result length**.

With that CODING is the ideal tool

- coding **voice over IP**, blue tooth/wireless data,
- coding **multimedia** information (e.g. MP3 data, DVD movies etc.),
- coding any **exchange of data** (e.g. between customer/client and supplier **in industry**) using an extreme uncertain and/or unsafe (public) connection (e.g. via **internet**).

1.1 Performance

Time-critical coding of mass data is a special challenge. This amongst other things arises in real-time scenarios, in which a continuous data stream has to be handled just-in-time. Any delay by data processing being too slow causes a collapse and therefore cannot be used, i.e. it results in suspending the ciphering process.

The method CODING reduces the amount of multiplications being necessary to a minimum being required for maintenance of complete safety even if data is of most homogeneous type (tested case: 1 GB binary zeroes). Because of this the referred processing speed being significant faster in comparison with other methods will be achieved, where the CODING-method being more flexible than other methods simultaneously gives a higher safety (especially for t>1) on the other hand.

1.2 Operation Safety

Conventional coding systems as DES, Triple-DES, AES at fist are working with fixed block length, so shorter data first has to be expanded to such fixed length. At second interlinked blocks are a necessity of such systems in many cases to get a sufficient safety, i.e. a defective block (e.g. by transmission error) makes the deciphering of all following blocks impossible.

CODING however has no fixed block length. I.e. shorter information at the end of a dataset is not expanded, so the length of the ciphered dataset is unchanged referring to the deciphered data.

For the operation safety more essential is the fact that a damaged block doesn't stop the deciphering of the following blocks. Especially this is relevant for telephone conversations (voice over IP), in which a trouble in data stream hasn't have the automatic consequence to put up the call completely new. Also transmitting other data with CODING, it is only necessary to send the incorrect block once again. By modifying the here presented algorithm (e.g. retaining the restart point parameters of a defective block), the incorrect blocks can be processed later in a very fast and efficient way.

1.3 Encryption Safety

Even newer coding systems as AES use only keys with maximum length of 2048 bits. In comparison already the simplest version of CODING (dimension t=1) uses a key with maximum length of 2048 bits (for dimension t=2 it already comprises 1,048,576 bits of maximum key length, for dimension t=3 it comprises 402,653,184 bits). In connection with the input of such keys, CODING includes a method to integrate datasets into the process of key generation by overlaying the key. Only this method facilitates the generation of useful "random" keys of the indicated sizes.

According to explorations of the author (http://eprint.iacr.org/2013/742.pdf) the method can't be broken without trying out the most part of all possible keys one after the other. An exact judgement of the safety especially in the view of information science (i.e. do you always need to check all possible keys for breaking a nontrivial key or can you reduce the number of possible keys to be checked) has to be left to a detailed examination.

Regardless of that the author has made extensive tests which prove that, even if data is of most homogeneous type (only binary zeroes), the method converges against the rectangular distribution of binary zeroes/ones and bytes or byte-groups respectively.

Further near key examinations have been able to show that even experimental keys differing form the original key in only one or few bits don't result in relevant statistical changes of distributions in relation to the coding data. Also inspecting block differences and its distributions, it couldn't discover relevant differences to block contents by chance.

Rather the method presented here can be said to be safe even if both the original data and the encoded data are known besides the coding algorithm. It is impossible to derive the original key from this information (except testing all possible keys).

Even compromising a block segment – using highest cipher state – doesn't compromise former or subsequent block segments automatically.

1.4 Future Safety

The method CODING covers a (infinite) number of concrete coding processes. The dimension t of a process within CODING is defined as the number of bytes combined to a byte-group. For t=1 single bytes are looked at and 1-byte substitution tables are used. For t=2 each 2 bytes are interpreted as a separate byte-group, for which 2-byte substitution tables are used, etc.

The method CODING with regard to t is **not limited**, at the moment PC-/Unix-programs fit for use exist until t=3 developed by the author.

1.5 Power of Key Spaces

If you "only" look at the power of the key space (in general (256**t)**(256**t) elements), for t=1 approx. 3.2317*10**616, for t=2 approx. 6.74114*10**315,652, for t=3 approx. 10**121,210,686 possible key space elements exist (x**t, say "x to the power of t", is defined as x*x*...*x with the value x appears t times). The last number is describable by a single 1 followed by zeroes which occupy more than 33 volumes with 1000 pages any of them with 3600 zeroes (in comparison: atoms in universe: ca. 10**80). It is easy to prove that at least (256**t)! elements of the key space can be really effective by the algorithms mentioned before (n!, say "n factorial", is defined as 1*2*3*...*n), actual all elements of a key space are effective.

Under these circumstances t has to choose only "big enough" to give nobody even with future conceivable (quantum) computers or computer farms any realistic possibility to decipher data enciphered by CODING without corresponding key or ciphering parameters in acceptable time. At the moment this should be the case for t=2 and bigger.

2 Characteristics

- § CODING is an „in place" method i.e. there is no change of data length.
- § CODING with high operation velocity is suitable for real time processes in stream cipher mode too.
- § CODING is usable for (nearly) any block length (also variable block length).
- § CODING represents
  - a very high operation safety,
  - an extreme high encryption safety,
  - an extensive protection against compromising, and
  - an excellent future safety.

3 Development Stage

The described method CODING has its beginning in preliminary examinations of the author more than 20 years ago. The consistent development, generalization, and implementation in Free Pascal programs up to now for the dimensions t=1, t=2, and t=3 recently have been driven on the basis of changed requirements on part of the potential users and the possibilities of computers on part of the implementation.

The CODING programs are **completely tested and fit for use** for t=1, t=2, and t=3.