Journal of Universal Computer Science, vol. 24, no. 1 (2018), 25-42
submitted: 25/5/17, accepted: 26/1/18, appeared: 28/1/18 © J.UCS

Identifying Encryption Algorithms in ECB and CBC Modes Using Computational Intelligence

Flávio L. de Mello
(Federal University of Rio de Janeiro, Brazil
fmello@poli.ufrj.br)

José A. M. Xexéo
(Military Institute of Engineering, Rio de Janeiro, Brazil
xexeo@ime.eb.br)

Abstract: This paper analyzes the use of machine learning techniques for the identification of encryption algorithms, from ciphertexts only. The experiment involved corpora of plain texts in seven different languages; seven encryption algorithms, each one in ECB and CBC modes; and six data mining algorithms for classification. The plain text files were encrypted with each cryptographic algorithm under both cipher modes. After that, the ciphertexts were processed to produce metadata, which were then used by the classification algorithms. The overall experiment involved not only a high quantity of ciphertexts, but also time consuming procedures for metadata creation as well as for identification. Therefore, a high performance computer and customized memory management were employed. As expected, the results for ECB mode encryption algorithm identification were significantly high, and also reached full recognition. On the other hand, algorithm identification under CBC is supposed to be marginal, but successful identification was up to six times higher than the probabilistic bid.

Keywords: cryptographic algorithm identification, data mining, machine intelligence, parallel computing
Categories: E.3, K.6.5, L.4.0

1 Introduction

This paper investigates ciphertext analysis for the identification of cryptographic algorithms used in encryptions using machine learning and data mining techniques. The aim is to use encrypted text files and a massive analysis to produce metadata in order to identify the cryptographic algorithm used for encryption, while at the same time employing machine learning techniques for classification. Generally speaking, cryptographic algorithms are essential for providing privacy, non-repudiation, integrity and authenticity, by ensuring only the sender and receiver are able to access the original information content. Kerckhoffs's assumption states that the security of a message must be based on key strength and not on the lack of knowledge about the features of the algorithm since such algorithms are considered known by everyone.

Public knowledge of algorithms notwithstanding, there are two possible scenarios for cryptanalysis. The first scenario is when it is possible to gather information about the cryptographic system. In this situation, the information on which cryptographic algorithm was used can be obtained by eavesdropping messages for algorithm negotiation, reading protocol specifications, performing social engineering, or even performing reverse engineering of hardware and software. The second scenario concerns digital forensics when the ciphertext is the only information available to the cryptanalyst. In this scenario, identifying which encryption algorithm is being used from the ciphertext alone is a challenging task. This is usually the case in attempts to collect evidence from computer media or network traffic seized at a crime scene.

The present work concerns the latter scenario, when a forensic cryptanalyst faces the challenge of knowing nothing about the cryptographic algorithm used. Therefore, the identification of the algorithm used is an important procedure because it facilitates obtaining the original data by cryptanalysis. Discovering which encryption algorithm was used is one of the activities that contribute to decoding, and it precedes key size determination and key discovery itself, as well as other cryptanalysis attacks.

When used to encrypt the same plain text with the same key, symmetric block ciphers in ECB (Electronic Codebook) mode generate identical encrypted text blocks. Despite this being an expected behavior, as is the clustering of ciphertexts according to such algorithms, several papers have been published expressing interest in the identification of cryptographic algorithms in ECB mode. Common sense states that encryption algorithms must generate sequences with random characteristics so that a ciphertext encrypted with algorithm A may be classified into a group of ciphertexts encrypted with algorithm B. However, several experimental studies have shown that this does not occur, and the groups generated in clustering processes are not mixed.

Mello et al. [Mello,16] show that, contrary to what one would expect from files encrypted via well known and widely used algorithms, there is enough exposed information to identify which cryptographic algorithm was used. The present article, in turn, is the result of ongoing research, it highlights some interesting observations and provides answers to some previously formulated questions. It has three major contributions. First, it presents a novel way to use encrypted text features during supervised training. Second, the amount of cryptographic algorithms to be analyzed is greater than that of previous investigations, since not only ECB encryption mode is addressed, but also CBC (Cipher Block Chaining) mode. In fact, this is the most important issue here, since CBC mode is not supposed to be sensitive to distinguishingattacks, which this paper shows not to be true. The CBC encryption mode may be susceptible to such an attack, but the volume of data to be processed is prohibitive for conventional computers, and it is needed parallel support from high performance computing architecture. Third, the paper describes the computational resources needed to fully identify CBC cryptographic algorithms.

2 Related Work

The attempt to acquire information about the cryptographic system used for generating a ciphertext is known as a “Distinguishing Attack”. One approach is to distinguish ciphers from random sequences. The other is to detect differences between ciphertexts generated by different algorithms. Maheshwari [Maheshwari,02] and Chandra [Chandra,02] tried to separate ciphertexts encrypted with DES from those encrypted with IDEA. Rao [Rao,03] used linear programming techniques in an attempt to separate RSA from IDEA. Carvalho [Carvalho,06] and Souza [Souza,07; Souza,08] grouped ciphertexts generated by RSA, DES and AES, using information retrieval techniques. Furthermore, Souza [Souza,07; Souza,03] described an application of neural networks to the same problem, but obtained less successful results than clustering techniques. Nagireddy [Nagireddy,08] developed histogram methods and block predictions to identify DES, AES, Blowfish, Triple DES and RC5. Saxana [Saxana,08] used machine learning techniques to classify ciphertexts generated by the algorithms Blowfish, RC4 and Camellia. Torres et al. [Torres,11] reported on the use of graph theory techniques and genetic algorithms for detecting patterns in ciphertexts. Souza and Tomlinson [Souza,03] presented a neural network capable of separating single 128-bit key ciphertexts produced by MARS, RC6, Rijndael, Serpent and Twofish. It should be noted, however, that all these works employed a distinguishing attack against a small set of algorithms, and they always considered only the ECB block cipher mode. There are two main reasons for these commonalities: (1) separating a small number of algorithms reduces the complexity involved in identification; and (2) the statistical behavior of sequences encoded in the ECB mode have smaller entropy than in CBC mode, although sometimes this cannot sensitize statistical test (see section 5 for details). The present work, on the other hand, analyzes a larger set of algorithms and also examines the CBC mode.

Progress in the use of computational intelligence to explore weaknesses in cryptographic systems has been gradual. Laskari [Laskari,07] used computer intelligence to perform a security check of pattern generation. Attempts to identify block ciphers have been carried out using neural networks [Albassal,04] and regression methods [Swapnaa,10]. The Rogers Isomorphism Theorem was used to demonstrate the biunivocal correspondence between a machines' understanding of text and the process of ciphertext analysis [Carvalho,16] based on knowledge geometry [Mello,15]. Ciphertext analysis was also performed using information retrieval [Carvalho,06; Souza,07; Nagireddy,08], an artificial intelligence technique that combines statistics and computational linguistics. These approaches suggest that a ciphertext may be considered a text in a language produced by the encryption algorithm and understanding the language depends on its linguistic structure. Moreover, evaluation of the encryption key [Souza,08] has also been analyzed using classificatory processes of computational intelligence. Lastly, cryptographic algorithm identification was investigated by combining intelligent classification processes with genetic algorithms [Xexeo,11] and employing statistical analysis [Sharif,10].

The concepts behind this paper are based on Barbosa et al. [Barbosa,16], who performed a preliminary experiment of very reduced scope to evaluate algorithm identification using the machine learning approach and to understand the technical constraints concerning computational resources. Subsequently, Barbosa et al. [Barbosa,17] applied the technique to multimedia files, and Mello et al. [Mello,16] used Barbosa's approach with a larger dataset and a greater number of algorithms to be identified. The present work presents new features of the metadata used by the machine learning techniques. Moreover, the most important and novel aspect of this paper is the analysis of not only ECB mode but also the CBC mode. Such an analysis suggests that distinguishing attacks against CBC block cipher mode are viable, and the computational resources required to obtain complete success in such a task are estimated.

3 Encryption Algorithm Identification

The strategy for determining the encryption algorithm used to produce a ciphertext consists of applying feature-identification methods to a set of attributes that describe the encrypted files. Although one could exploit the same algorithms to perform data mining, in this work such algorithms were exploited to perform pattern recognition, and therefore perform machine-learning. Figure 1 shows the schematics, which is composed of three main modules. The first module is responsible for taking the corpora of plain texts and combining it with keys and encryption algorithms in ECB and CBC modes, thereby producing the ciphertexts. The next module takes the ciphertexts and carries out a transformation, resulting in metadata that describe features of those encrypted files. Finally, the last module involves machine-learning methods that exploit the large amount of metadata with the purpose of identifying the encryption algorithm.

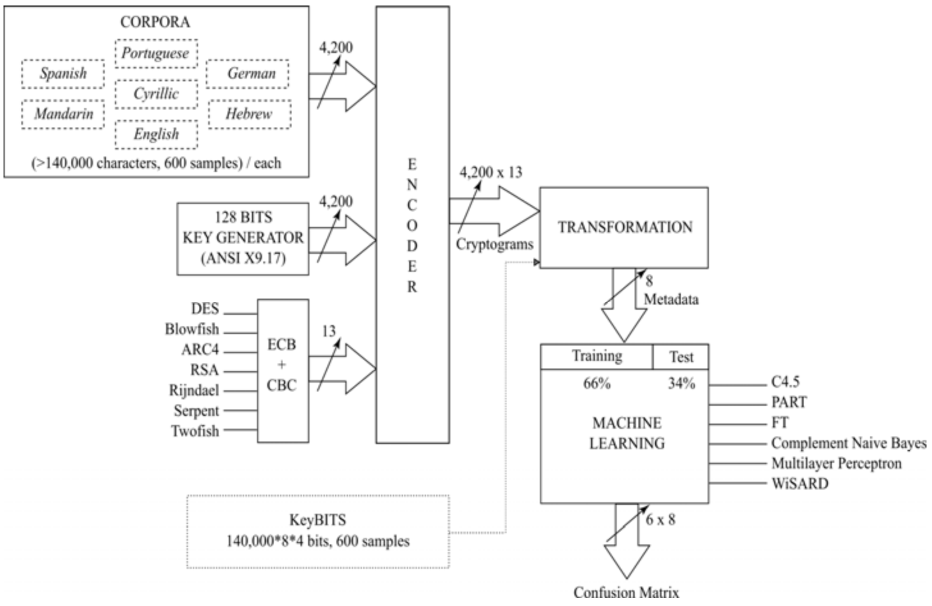

Figure 1: Schematics for determining the encryption algorithm used to produce ciphertext.

The corpora of plain texts used in this research included seven different languages. It seemed interesting to evaluate whether the original vernacular could influence the encryption algorithms, and if such an influence could be captured by the classification algorithms. The original idea was to use Portuguese, Spanish, English and German texts. However, there was an opportunity to enlarge the number of writing systems, and since this would improve the analysis of how cryptographic algorithms increase entropy, it seemed reasonable to enhance the sample space by incorporating other languages, such as Hebrew, Cyrillic and Mandarin.Moreover, these additional texts could not be encoded in ASCII, but in Unicode, which implies an increase in bytes used for character representation.

The corpora were composed of 4,200 text samples divided into seven distinct corpora. Each corpus corresponded to a language system and consisted of 600 samples of different texts retrieved from newspapers and magazines, with no repeated sentences. Each sample had at least 140,000 characters, which corresponds to approximately 35 text pages, an unusual size for newspaper and magazine articles. Thus, full text articles were concatenated until the minimum number of 140,000 characters was reached.

It should be noted that in order to avoid this overhead in gathering small text and merging them into a single text, it is common to use long texts for corpus assembly, such as the Bible, the Iliad, the Odyssey, and Don Quixote. However, the linguistic construction of these works is so distinct from everyday texts that they do not provide a good representation of contemporary languages. This may introduce a bias into the sample space, so the use of texts from newspaper and magazine articles has become a mandatory requirement.

Since this study is exploratory in nature, classical cryptographic algorithms were chosen for evaluation. The criterion is based on evaluating the behavior from block encryption, stream cipher encryption, symmetric and public key algorithms. Thus, the selected cryptographic algorithms were DES, Blowfish, RSA, ARC4, Rijndael, Serpent and Twofish. The implementations of these algorithms was based on source codes from Schneier [Schneier,96] and Dai [Dai,16], but each implementation not only had to be adapted to the execution environment, but also had to be modified to allow its application in ECB and CBC modes, thus resulting in 13 implementations (recall that ARC4 is a stream cipher encryption). An important remark must be made concerning RSA since it is not a block cipher algorithm, and thus ECB/CBC modes are not applicable. In this paper, the traditional algorithm is called RSA, while RSA "CBC" describes a variant implementation where the text is broken up into separate blocks, and an "add mod N" is performed of the ciphertext of one block to the plaintext of the next block, almost like CBC mode. Finally, initialization vectors of all CBC algorithms were produced by a pseudo-random bit generator derived from the ANSI X9.17.

Parallelism was obtained by using OpenMP [OpenMP,11] directives in order to signal to the compiler which regions contained independent iterations, and thus liable to be parallelized, in order to take advantage of a high-performance computer. The precise gain obtained through the parallelism of those algorithms extrapolates the primary purpose of this work, but prototype testing suggests a significant advantage that will be discussed in Section 4. Of course, this benefit occurs with parallel implementations of algorithms in ECB mode, which is nonexistent in CBC mode. This is because of the high dependence of CBC mode loop iterations, which computes an XOR between the current block cipher and the block cipher from the previous iteration.

All encryption keys were composed of 128 bits and were produced by a pseudo-random bit generator (PRNG) derived from ANSI X9.17 [Schneier,96], with two exceptions. Since a DES key needs just 56 bits, the 128-key needed to be shortened. Additionally, it is well known that RSA needs two large prime numbers as a private key, but these numbers can be substituted by the output from a secure pseudo-random number generator. The RSA' public and private keys are produced by concatenating 8 blocks of 128 bits from the PRNG. Each plain text file from the corpora was associated with a different key in order to avoid any influence of key patterns on the data mining process. The usage of the same key to encrypt distinct files may induce a defect in the cryptographic algorithm identification process, which is subject to being captured by the classification algorithms, consequently masking results. This kind of key generation does not allow operation parallelism, but this task is not computationally expensive.

Once encoding was finished, metadata creation was initiated. This procedure extracts useful information from the ciphertexts that is employed in the machine learning stage. Each metadata file was associated with one of the seven language corpora and with all cryptograms produced from that corpus. An additional metadata file was also created corresponding to the fusion of all encoded files, from all languages, into a single corpus. The aim of this extra file was to evaluate the behavior of identification algorithms when applied to a dataset composed of all seven languages. The transformation module also received 600 samples from another module called KeyBITS; it is not part of the identification process, and its purpose will be explained in the next section.

Each line record from a metadata file is composed of a histogram from a cryptogram, whose bins are the occurrences of a block of contiguous bits, and a label for cryptographic algorithm designation. This label is used for training the machine learning algorithms and as data control for evaluating results. The bit size of these blocks varied from 2 to 34 bits. The 34-bit upper limit was imposed by memory capacity as a consequence of resource availability in the parallel computer and by the implementations of the data processing modules and machine learning techniques. There is an important issue concerning the block collection of contiguous bits used by Mello et al. [Mello,16] and the one from this paper. The former isolated blocks according to contiguous bit size, and so the machine-learning algorithms analyzed them separately. The present study, in turn, grouped all blocks into a single line record in order to analyze them together, and eventually explore relationships between different block sizes. Moreover, the OpenMP directives were used to signal to the compiler what commands from bins calculus can be parallelized. The parallelization of this transformation stage is extremely necessary because without it the time for computing all metadata becomes prohibitive.

Finally, the metadata were processed by a set of classifiers at the machine-learning module. As previously mentioned, this stage uses the bin information of histograms cumulatively. The usual approach in the literature is to use a given bin size information to create the learning model and perform the classification. However, in the present work, all previous bin size information was used to execute these tasks. That is, when computing the learning model and performing the classification, at the iteration whose bin size is S, all information from bins of size 2 to S-1 were also used.

Like the cryptographic algorithms, the criterion for choosing the classification algorithms was to explore the viability of using machine learning algorithms in cryptographic algorithm identification. Thus, algorithms were used that represent various categories of classifiers, such as bayesian, functional, rule-based, and decision tree generators. For each algorithm, independent iteration loops were located and OpenMP directives applied in order to explore parallelism in order to reduce computation time.

The classification algorithms used were: C4.5, PART, FT, Complement Naive Bayes, Multilayer Perceptron and WiSARD. Although these algorithms are classical, they are not outdated or obsolete. Brand new machine learning algorithms provide slightly better accuracy, are usually faster and consume less memory, but they are not easy to obtain for the target platform being in this experiment. These features may reduce the demand for computational resources, but do not contribute significantly to improved accuracy, which is the main need of the present work. Thus, availability becomes the major criterion for choosing machine-learning methods, and the classical algorithms surpass the new ones in this regard.

The machine learning procedure was divided into two sequential steps; one concerns the creation of the classification model for each classifier, and the other consists of testing the classification. Therefore, each metadata file was divided into two distinct parts. The first, containing 66% of all cryptograms, was used to assemble the classification model. The second, comprising the remaining 34% of all cryptograms, was used for testing. It is important to note that the experiment maintained the same amount of metadata ciphertexts for each cryptographic algorithm and for both ECB and CBC modes, so that classification algorithms were neither over-trained nor under-trained for identification. There was also a similar balance for each language with the metadata file that joins all languages. Consequently, this procedure removed the possibility of favoring the identification of one algorithm over another, as well as any influence of a better-trained language set. The machine-learning module produced a confusion matrix for each classification algorithm and each metadata file. The goal was to acquire an effective measure of the classification model by presenting the number of correct classifications versus the predicted ones. From these confusion matrices it was possible to obtain the percentage of correct classifications for each classifier, for each algorithm and for each language corpus.

4 Description of the Experiment

For the experiment, each classifier was trained using 10-fold cross-validation. The C4.5 algorithm was configured to use just one boosting interaction, starting with an empty weights vector and with no global pruning step to simplify the tree. For the FT algorithm, the minimum number of instances for a node to be considered for splitting was set to 15, the number of iterations for LogitBoost was set to 30, and no weight trimming. For the PART algorithm, the confidence factor used for pruning was set to 0.25, the minimum number of instances per rule was set to 2, and 3-folds was used for error pruning. The Complement Naive Bayes algorithm does not implement Laplace smoothing and density was used as metric predictor. The Multi Layer Perceptron algorithm was initially configured with randomized weights, the learning rate was set to 0.005, the momentum was set to 0.2, there were two fully connected hidden layers with 350 neurons, and the calibration used back propagation. Finally, the WiSARD algorithm was configured with 13 RAM-discriminators, and the bleaching technique was used with a confidence value of 0.1.

The environment in which the experiment was developed was a Cray XK6 Nano parallel computer. This machine has 40 computational nodes, each with a 16-core Opteron 6276 CPU, a NVIDIA Tesla K20 GPU and 32GB of central memory. The experiment, however, used only the 16 cores from one computational node in order to avoid parallel constraints concerning CPU and GPU communication as well as inter-node communication. Moreover, the experiment was checked twice on an Environment equipped with 4 nodes of Intel Xeon Phi CPU 7250 @ 1.40GHz, 68 cores/272 threads each, and 128GB central memory

Barbosa et al. [Barbosa,16] had previously pointed out the demand for computing power in a significantly smaller experiment, as have other researchers [Mello,16] [Barbosa,17]. In order to obtain greater computing ability, it was necessary to use not only a machine with high computing power but also parallel programming. A comparison of algorithm processing time between the present experiment and that of Barbosa et al. [Barbosa,16] indicates an increase in speed of 864.02. Moreover, the increased speed of the parallelized algorithm using 16 cores versus using only one core was 14.24. Table 1 presents the processing times for each module of the experimental scheme, and clearly shows that the transformation module and the data-mining module have a high cost. It should be noted that the experiment was not completely automated and the values of Table 1 represent the actual computation time, and does not take into consideration manual activities to maneuver data from one module to another.

Module	*Execution (dd:hh:mm:ss)*	
Key Generator	00:00:00:08.904	
Encoder	00:01:52:40.621	
Transformation	09:19:54:11.826	
Machine Learning	30:13:51:51.885	
Classifier	*Train*	*Test*
C4.5	00:11:18:45.026	00:02:24:21.614
FT	02:18:58:05.186	01:07:12:19.338
PART	00:10:27:32.025	00:05:20:37.937
Complement Naive Bayes	00:00:10:59.461	00:00:00:48.647
Multilayer Perceptron	24:06:11:41.931	00:00:29:42.673
WiSARD	00:23:04:21.649	00:00:12:36.398

Table 1: Experiment processing times.

The procedures for transforming ciphertexts into metadata is costly due to the large number of bins in the histograms, which is directly proportional to number of bits used to define block size. In the worst case, the 34-bit block had 2^{34} classes, which demands the construction of a dynamic data structure (hash table chained with linked lists) for storing the number of times a given bin has occurred in the data set. It was not possible to increase this number of bits due to RAM constraints of the available environment; however a better data structure can mitigate this problem.

The machine learning stage was also quite long and the Multilayer Perceptron classifier was a key factor in this result, as expected. For this reason, the WiZARD neural network was also evaluated as a possible replacement for Multilayer Perceptron, since the former is known to be faster than the latter. Furthermore, the computational cost of the classifier Complement Naive Bayes must be highlighted, since it was significantly lower than all the other classifiers.

5 Results

Randomness tests were first performed to analyze the distribution of the encryption keys and the individual ciphertexts in order to determine if there are patterns that might confuse further analysis. Tests are usually grouped into test batteries (or test suites), which provide more complex randomness analysis. Passing such tests is an important step for a pseudo-random sequence to be approved by certification authorities.

The randomness test from NIST (National Institute of Standards and Technology) Statistical Test Suite [Rukhin,10] was applied to 600 sequences of cryptograms and to the encryption key file in order to detect any grossly inadequate random number generators, some hardware and software implementation errors, and normal imperfections of pseudo-random number generators. This suite defines 15 empirical tests of randomness, some of which are divided into sub tests. The suite report indicates the number of failures obtained during such an evaluation. Table 2 summarizes the results of these tests for the cryptograms of the present work.

Proportional failure (Prop) occurs when a generator fails a type of test too often. The NIST guidelines define a tolerance, which is a level of confidence beyond which a sample of generator output can be said to have not come from a true random source. None of the cryptographic algorithms of the present study were rejected, nor the ANSI X9.17 key generator.

Furthermore, a single test is considered passed if the P-value is above the significance level of 0.01 or below 0.99. The P-value represents the probability of a perfect random number generator producing the same or worse test result. Table 2 presents some small failures, but the interpretation of these is not straight-forward since the probability of a good random number generator failing this requirement is not zero. Thus it is not surprising for occasional P-values to be smaller than 0.01, such as for some instances reported in Table 2. Besides, the smallest P-value obtained was 0.004462 (ARC4), which is not a big failure, but it is known that this algorithm fails statistical tests [Goutam,07]. Therefore, the randomness test is in accordance with what is expected from cryptographic algorithms, and thus the individual analyses of the ciphertexts do not indicate deviation from randomness.

The next aspect analyzed was the influence of the original language of the texts on cryptographic algorithm identification. The confusion matrices make it possible to determine the percentage of correct identifications for the seven given language corpora (C_L) and for the additional corpus containing all languages (C_{ALL}). Considering all bin sizes, differences between the percentages for the individual single language corpora and the joint language corpus were calculated as the normalized distance $D=(C_{ALL}- C_L)*100/ C_{ALL}$. Figure 2(a) represents these distances for the correct identification of the DES encryption algorithm by the C4.5 classifier, where the mean value is 3.014 and the standard deviation is 2.347. Moreover, Figure 2(b) shows the distances for the RSA encryption algorithm and the Complement Naive Bayes classifier, with a mean value of 3.004 and a standard deviation of 2.496.

Empirical Test		**Cryptograms** (First column ECB mode, second column CBC mode – except for ARC4; see comment in section 4 regarding RSA.)												
		ARC4	**Blowfish**		**DES**		**Rijdael**		**RSA**		**Serpent**		**Twofish**	
Frequency	*P-val.*	0	0	0	0	0	0	0	0	0	0	0	0	0
	Prop.	0	0	0	0	0	0	0	0	0	0	0	0	0
BlockFrequency	*P-val.*	0	0	0	0	0	0	0	0	0	0	0	0	0
	Prop.	0	0	0	0	0	0	0	0	0	0	0	0	0
Cumulative Sums (+2)	*P-val.*	0	0	0	0	0	0	0	0	0	0	0	0	0
	Prop.	0	0	0	0	0	0	0	0	0	0	0	0	0
Runs	*P-val.*	0	0	0	0	0	0	0	0	0	0	0	0	0
	Prop.	0	0	0	0	0	0	0	0	0	0	0	0	0
LongestRun	*P-val.*	0	0	0	0	0	0	0	0	0	0	0	0	0
	Prop.	0	0	0	0	0	0	0	0	0	0	0	0	0
Rank	*P-val.*	0	0	0	0	0	0	0	0	0	0	0	0	0
	Prop.	0	0	0	0	0	0	0	0	0	0	0	0	0
FFT	*P-val.*	0	0	0	0	0	0	0	0	0	0	0	0	0
	Prop.	0	0	0	0	0	0	0	0	0	0	0	0	0
Non-overlapping Template (+148)	*P-val.*	2	2	1	2	1	0	0	2	1	1	1	1	0
	Prop.	0	0	0	0	0	0	0	0	0	0	0	0	0
Overlapping Template	*P-val.*	0	0	0	0	0	0	0	0	0	0	0	0	0
	Prop.	0	0	0	0	0	0	0	0	0	0	0	0	0
Universal	*P-val.*	0	0	0	0	0	0	0	0	0	0	0	0	0
	Prop.	0	0	0	0	0	0	0	0	0	0	0	0	0
Approximate Entropy	*P-val.*	0	0	0	0	0	0	0	0	0	0	0	0	0
	Prop.	0	0	0	0	0	0	0	0	0	0	0	0	0
Random Excursions (+8)	*P-val.*	0	0	0	1	0	0	0	0	0	0	0	0	0
	Prop.	0	0	0	0	0	0	0	0	0	0	0	0	0
RandomExcursionsVariant (+18)	*P-val.*	0	1	0	0	0	0	0	0	0	0	0	0	0
	Prop.	0	0	0	0	0	0	0	0	0	0	0	0	0
Serial (+2) .	*P-val.*	1	0	0	0	0	0	0	0	0	0	0	0	0
	Prop.	0	0	0	0	0	0	0	0	0	0	0	0	0
Linear Complexity	*P-val.*	0	0	0	0	0	0	0	0	0	0	0	0	0
	Prop.	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 2: Number of failures reported by NIST randomness test.

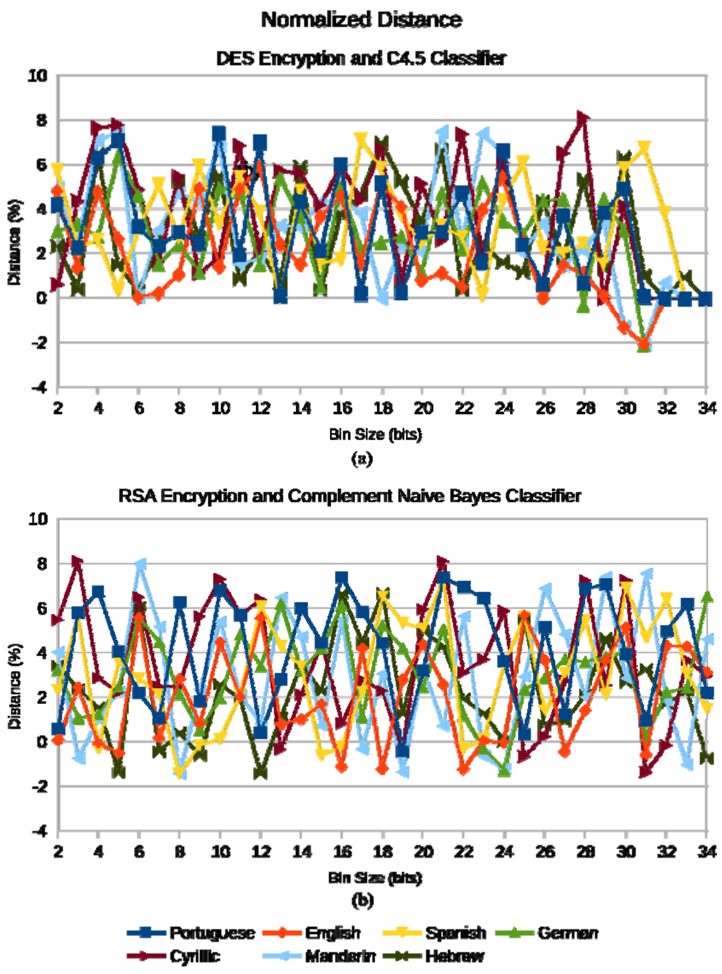

Figure 2: Differences in successful identification between a single language corpus and the seven language corpus: (a) distance for DES encryption analyzed by the C4.5 classifier, (b) distance for RSA encryption analyzed by the Complement Naive Bayes classifier.

Positive values of *D* indicate that the best identification results were obtained when using the corpus with all languages together. In the vast majority of cases this is true, but the nominal values obtained for mean distance do not indicate a significant difference. The standard deviation also does not show an important dispersion of the values of the data set. Therefore, the original language of the plain texts does not seem to interfere with the performance of the classification algorithms. The better results obtained for the corpus with all languages together is explained by its sample space size. Since it has many more instances, the classification model for the corpus with all languages is finer trained, and hence produces better results. Consequently, all further analysis in this paper corresponds to the corpus with all seven languages together.

The KeyBITS block in Figure 1 corresponds to a physical pseudo-random bit generator [Barbosa,16b], which uses the intrinsic light noise of a laser beam as an entropy source. The purpose of using KeyBITS is to create 600 samples of pseudo-random files of sizes equivalent to those from the ciphertexts. Notice that this random bits collection may not be truly random, but its entropy is high enough to be considered so [Barbosa,10]. The assumption made is that these files do not have exposed patterns that allow successful algorithm identification by the machine learning algorithms. Thus, the KeyBITS files were inserted into the identification procedure in order to produce base line values and serve as a reference for comparison. The machine learning algorithms of the present investigation belong to a category of classification algorithms, which means they are employed when there is a set of predefined classes and the intention is to know which class a new object belongs to. This means that the classification algorithms will not be able to identify KeyBITS since it cannot be described by patterns. On the other hand, if the machine learning algorithms were clustering algorithms, that is, if such algorithms attempt to group a set of objects and find whether there is some relationship between them, then there would be a very high hit since the KeyBITS group is defined by all elements that do not belong to the other encryption groups.

The next aspect analyzed was the identification performance of the classifiers. Note that the probabilistic control value for a correct random identification of an encryption algorithm is 7.14% (100/(13+1)), since the sample space contains equal amounts of metadata for the 13 cryptographic algorithms plus KeyBITS. However, if just the high entropy instances are considered (BlowFish, Twofish, Serpent, Rijndael and DES in CBC mode; RSA “CBC”; KeyBITS) the probabilistic control value is 14.29% (100/7) because they are the only ones to pose a challenge to the identification process. Figure 3 presents six graphs, one for each classification algorithm. At first inspection the graphs immediately call attention to the existence of two distinct behavioral groups. The top series in the charts, with better identification responses, correspond to the cryptographic algorithms in ECB mode and ARC4. On the other hand, the bottom series in the charts correspond to CBC mode algorithms with lower identification performance. In fact, the difference between these two groups is easily understood since the CBC mode increases entropy to higher levels than the ECB mode, thus providing a more difficult environment for identification. Despite this expected separation, it is still important for cryptanalysis that such a detachment did occur. In the following discussion, this paper will develop separate analyses for each block cipher mode. Finally, KeyBITS was not classified at all, as expected.

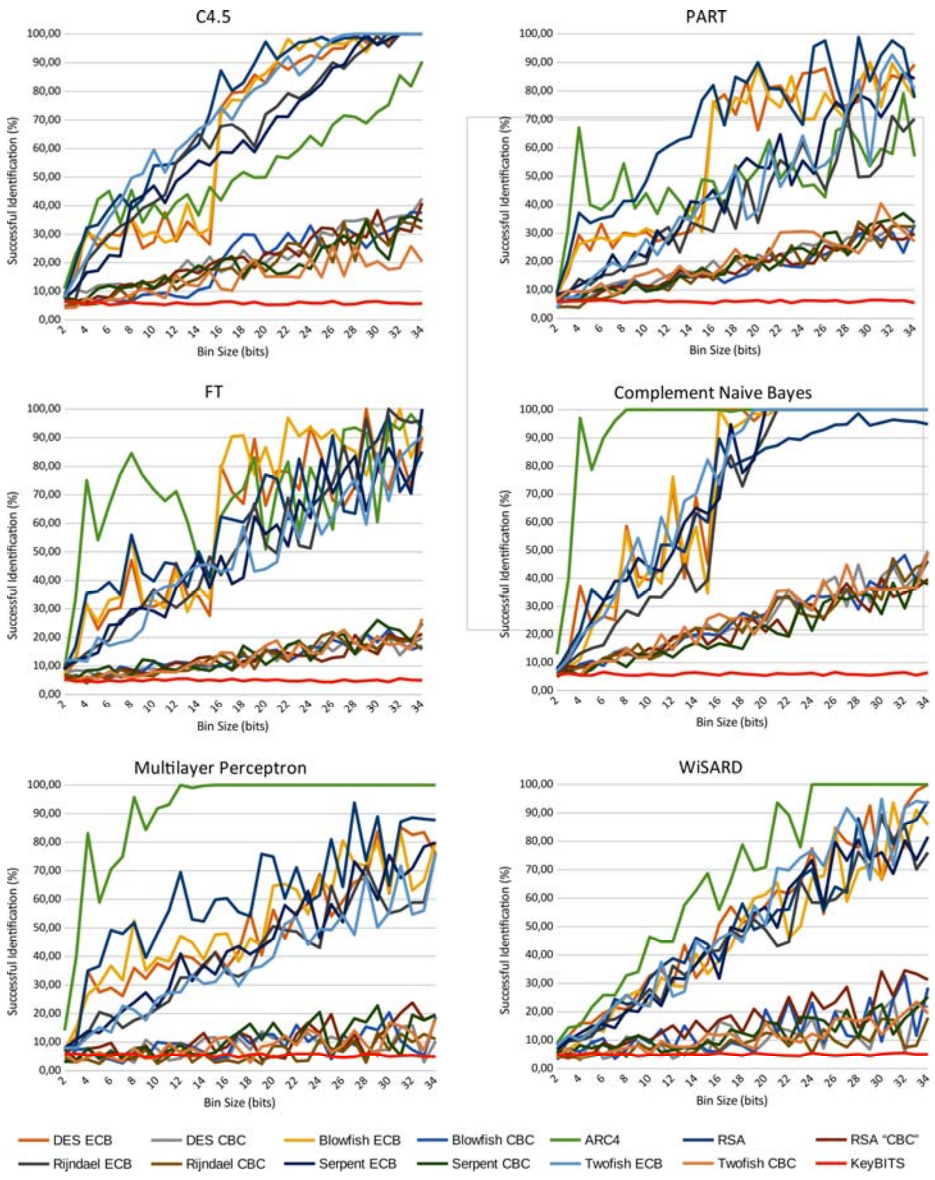

Figure 3: Success rate in the identification of cryptographic algorithms for classifiers C4.5, PART, FT, Complement Naive Bayes, Multilayer Perceptron, WiSARD and KeyBITS.

Considering the ECB mode group (plus RSA and ARC4), the experiment made an exhaustive search for possible outlines that may allow the identification of cryptographic algorithms. This procedure was expected to be successful in some cases and not in others, since the machine learning algorithms have different approaches to the same problem. It can be seen that the identification rates in the C4.5 chart converge to almost 100% at a 28-bit bin size, excluding ARC4. The charts for PART and FT do not exhibit such a convergence, but they still suggest an increasing ability for identification. Moreover, in these three initial charts there is a significant increase in the successful identification rate of DES and Blowfish when the number of bits used becomes larger than 16 bits. The Complement Naive Bayes chart exhibits the best results, with it being able to identify all algorithms with 100% accuracy (except RSA) when the bin size was greater than 20 bits.

This chart also shows a qualitative leap in identifying DES, Blowfish and Rijndael when the bin size exceeds 16 bits. The Multilayer Perceptron and WiSARD charts also exhibit increasing abilities for identification, with the latter being slightly better than the former. All of these features sustain Complement Naive Bayes as the classifier that provides the best results for identifying cryptographic algorithms in ECB mode. These results reinforce those reported by Mello et al. [Mello,16], even though in the present experiment the bin collection of contiguous bits are combined (see section 3), by producing identification profiles that are similar, but not equal, to the those previously reported. Furthermore, the identification of ECB mode encryption algorithms was significantly high because encrypting the same data block of plaintext using ECB mode always yields the same block of ciphertext, that is, repetitive sequences of bits in the plaintext result in repetitive patterns in the encrypted output, thus providing an opportunity for identification. This means that ECB propagates frequencies from plaintexts to ciphertexts and produces ciphertexts of non-uniform distribution, which makes classification easier.

On the other hand, the major result of this experiment was with regard to the CBC mode group; that is, the successful distinguishing attack against the implemented algorithms in CBC mode. CBC encryption mode is not supposed to be sensitive to statistical attacks, but the results shown in Figure 3 are evidence that it cannot be take unconditionally. Even though high values of correct identification were not achieved, there was an unexpected behavior. The CBC mode is known for increasing entropy to high levels and increasing randomness, thus, successful identification would not be expected to increase monotonically. However, except for Multilayer Perceptron, all classifiers exhibited increased identification performance as bin size increases. In addition, Complement Naive Bayes exhibited the greatest slope for the identification series.

There are some possible reasons why this happens. First, the patterns being identified may be the ones disguised at the first 128 bits from ciphertexts. Note that the first 128-bits block from CBC mode are encrypted just after an XOR with the initialization vector. This procedure may provide a weak random input blocks before encryption, compromising the first 128 bits. If this occurs, the remaining bits from the ciphertext are irrelevant for identification, no matter their entropy is. Second, if the cipher block is n bits long, then it should be expect a collision after $2^{n/2}$ encrypted blocks. However, there is a non-zero probability of such collisions happen before this number of encrypted blocks. Moreover, some artifact may be interfering with the experimental results increasing the probabilistic occurrence of such collisions. Third, some of the underlying ciphers may be vulnerable against distinguishing attacks (or perhaps the implementation) facilitating identification.

Since the cryptograms passed the NIST test, these results suggest that high entropy produced by the implemented algorithms is not sufficient for avoiding cryptographic algorithm identification by machine learning procedures. Moreover, the physical random bit generator KeyBITS, which was used as reference for comparison, was mixed among the other encryption algorithms with marginal successful identification very close to the probabilistic bid. This means that the classifiers did not manage to categorize a uniform distribution of bits, but they did manage to obtain a certain amount of success with the CBC distribution of bits produced by our implementation. This is a major result because it implies that it may exists a distinguishing attack against the CBC block cipher mode of operation, or at least, against the implementations used in this work.

The successful identification of cryptographic algorithms in CBC mode, when using Complement Naive Bayes classifier and considering a 34-bit bin size cumulatively, ranges 40-50%. This is in stark contrast to the 7.14% of the probabilistic bid and suggests that it is possible to identify algorithms in CBC mode. These series have a monotonic increasing behavior for successful identification, and thus it is convenient to study this property. Note that there are two possible scenarios. In the first, the monotonic increasing behavior experiences a reduction in slope and adopts an asymptotic course. In such a case, the data obtained from the experiment do not contribute to determining its saturation value. In the second scenario, the monotonic increasing behavior maintains its slope until full recognition. Thus, it is possible to model the relationship between the scalar dependent variable (successful identification) and the independent variable denoted by the bin size. Figure 4 shows a linear regression of these values.

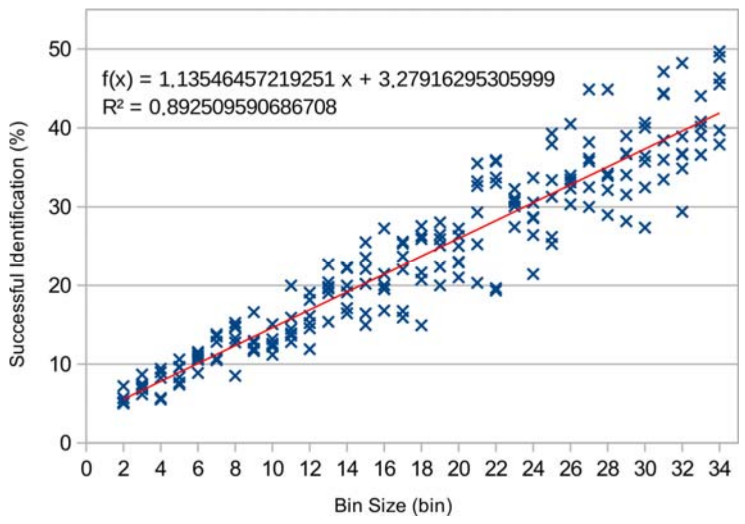

Figure 4: Linear Regression of the successful identification by the Complement Naive Bayes classifier.

Figure 4 also presents the linear regression equation *f(x)* and the coefficient of determination R^2. The R^2 value is near 1, and thus suggests a good fit for the model.

For this reason, it is interesting to extrapolate to predict the bin size for possible full recognition. From this point of view, the bin size value for full identification would be 85.18 bits, which implies a computational challenge. The number of bin variations for such a histogram is 2^{86}, a significantly higher number than the actual 2^{34} of this experiment thus demanding much more computational power and addressing capacity. This value is beyond the borderline of processing, that is, reaching this number of bin variations is solvable but not in an effective manner. Despite this problem is computable, it is not feasible, except if a new method is found, with smaller computational complexity.

6 Conclusion

This paper presented an experiment involving the use of machine learning for the identification of encryption algorithms. It demonstrated that the influence of the idiom in which plain texts were written is not relevant to the overall process. The NIST test was applied to cryptograms to guarantee the quality of the encoded texts. The proposed procedures obtained full identification for almost all of the selected cryptographic algorithms in ECB mode. However, the most important result was the identification of algorithms in CBC mode. Although the identification of algorithms in CBC mode exhibited lower rates than ECB, they were are not insignificant since they were greater than the probabilistic bid. Moreover, these rates increased monotonically, and thus can be increased by intensive computation. Lastly, the most efficient classifier was Complement Naive Bayes, not only with regard to successful identification, but also in time consumption.

It is suggested that future investigations increase the number of cryptographic algorithms analyzed. Considering the classification methods, the SVM algorithm was tested and produced regular results. However, it seems that it is still possible to optimize the parameters of SVM to achieve improved results. Moreover, it was not possible to increase bin size due to RAM constraints in the available environment, so the implementation of a more refined data structure is recommended in order to mitigate this problem. Furthermore, considering the high performance computing environment employed, it is important to improve the usage of the processing architecture by dispatching tasks not to just one computing node, but to all nodes and GPUs available, which would enhance computational capacity. Finally, the number of bins estimated to fully detect CBC mode algorithms could be reduced if it were possible to enrich metadata files with some kind of additional information.

Acknowledgements

This research received support from Poli 19.257 of Coppetec Foundation at Federal University of Rio de Janeiro (UFRJ). The work was developed at the Machine Intelligence and Computer Models Laboratory of Polytechnic School (IM^{2}C/Poli/UFRJ). The access to the high performance computer machine was granted by the Brazilian Cyber Defense Command (ComDCibers), the Military Institute of Engineering (IME) and the National Network for Information Security and Cryptography (RENASIC). Thanks for AI2Biz Lab, the Intel Center of Excellence for Artificial Intelligence, one of the Intel AI Builders Program partners, who allowed us to run confirmation tests at their computational environment. We also thank Sergio Genaro, Sergio Barbosa Villas-Boas (*in memorian*), Dan Castro Blumfeld and Alexandre Vieira Butman for their help in building the corpora of languages the authors do not understand. Thanks go to Eduardo Marsola do Nascimento for running the NIST test. We thank Dr. Erik Russell Wild, American biologist from University of Wisconsin, for providing a native English speaker revision of the manuscript.

References

[Albassal,04] Albassal, Ayman M. B.; Wahdan, Abdel-MoneimWahdan A. Neural network based cryptanalysis of a feistel type block cipher, International Conference on Electrical, Electronic and Computer Engineering, 2004, pp. 231–237, 2004.
doi: 10.1109/ICEEC.2004.1374430

[Barbosa,10] Barbosa, Geraldo Alexandre. Fast multi-photon key distribution scheme secured by quantum noise, US Patent 7,831,050 B2, 2010.

[Barbosa,16] Barbosa, Flávio Mendonça; Vidal, Arthur Reimão Santos Figueiredo; Mello, Flávio Luis de. Machine Learning for Cryptographic Algorithm Identification, Enigma - Brazilian Journal of Information Security and Cryptography, v.3, n.1, pp.3-8, 2016.

[Barbosa,16b] Barbosa, Geraldo Alexandre. A Wireless Physically Secure Key Distribution System, Enigma - Brazilian Journal of Information Security and Cryptography, v.3, n.1, pp.9-14, 2016.

[Barbosa,17] Barbosa, F. M.; Vidal, A. R. S. F.; Almeida, H. L. S.; Mello, F. L. Machine Learning Applied to the Recognition of Cryptographic Algorithms Used for Multimedia Encryption. Revista IEEE América Latina, 2017. doi: 10.1109/TLA.2017.7959350

[Carvalho,06] Carvalho, Carlos A. B. The usage of information retrieval techniques for cryptoanalysis/O uso de técnicas de recuperação de informações em criptoanálise. Master dissertation, Computer and System Engineering Department. Rio de Janeiro: Military Institute of Engineering - IME, 2006. (in portuguese)

[Carvalho,16] Carvalho, Roberto Lins de; Mello, Flávio Luis de. Rogers' Isomorphism Theorem and Cryptology Applications, IEEE Latin America Transactions, v.13, n.6, pp.2009-2016, 2015. doi: 10.1109/TLA.2015.7164229

[Chandra,02] Chandra, Girish. Classification of Modern Ciphers. Master Thesis, Indian Institute of Technology Kanpur, Kanpur, Indian, 2002.

[Dai,16] Dai, Wei. Crypto++ Library 5.6.3, 2015. <https://cryptopp.com/>, access may 1st, 2016.

[Goutam,07] Goutam, P.; Subhamoy, M. RC4 State Information at Any Stage Reveals the Secret Key, IACR Cryptology ePrint Archive, 2007.

[Laskari,07] Laskari, E. C.; Meletiou, C.; Stamatiou, Y. C.; Vrahatis, M. N. Cryptography and Cryptanalysis through Computational Intelligence. Computational Intelligence in Information Assurance and Security. Studies in Computational Intelligence, Springer Berlin Heidelberg, v.57, pp.1-49, 2007. doi: 10.1007/978-3-540-71078-3_1

[Maheshwari,2002] Maheshwari, Pooja. Classification of Ciphers. Master Thesis, Indian Institute of Technology Kanpur, Kanpur, India, 2002.

[Mello,15] Mello, F. L. de; Carvalho, R. L. de. Knowledge Geometry. Journal of Information & Knowledge Management, v. 14, p. 1550028, 2015. doi: 10.1142/S0219649215500288

[Mello,16] Mello, Flávio Luis de; Xexeo, Jose Antonio Moreira. Cryptographic Algorithm Identification Using Machine Learning and Massive Processing. Revista IEEE América Latina. , v.14, p.4585 - 4590, 2016. doi: 10.1109/TLA.2016.7795833

[Nagireddy,08] Nagireddy, Sreenivasulu. A Patternn Recognition Approach to Block Cipher Identification. Master dissertation, Department of Computer Science and Engineering, Indian Institute of Technology Madras, October, 2008.

[OpenMP,11] Open.Org. OpenMP 3.0, 2011. < http://openmp.org/ >, access may 1st, 2016.

[Rao,03] Rao, M. B. Classification of RSA and IDEA Ciphers. Master Thesis, Indian Institute of Technology Kanpur, Kanpur, India, 2003.

[Saxana,08] Saxana, G. Classification of Ciphers using Machine Learning. Master Thesis, Indian Institute of Technology Kanpur, Kanpur, India, 2008.

[Schneier,96] Schneier, Bruce. Applied Cryptography: Protocols, Algorithms, and Source Code in C, Wiley, 2 ed., 1996.

[Sharif,10] Sharif, S. O; Kuncheva, L. I.; Mansoor, S. P. Classifying encryption algorithms using pattern recognition techniques, Information Theory and Information Security, 2010 IEEE International Conference on, Beijing, pp. 1168-1172, 2010. doi: 10.1109/ICITIS.2010.5689769

[Souza,07] Souza, William Augusto Rodrigues de. Pattern Recognition in ciphertexts using text classification techniques/Identificação de padrões em criptogramas usando técnicas de classificação de textos. Master dissertation, Computer and System Engineering Department. Rio de Janeiro: Military Institute of Engineering – IME, 2007. (in portuguese)

[Souza,08] Souza, William Augusto Rodrigues de; Xexeo, José Antonio; Oliveira, Cláudia. Método de Agrupamento de Criptogramas em Função das Chaves de Cifrar, IV Workshop em Algoritmos e Aplicações de Mineração de Dados, Campinas, 2008. (in portuguese)

[Souza,03] Souza, W.A.R. AND Tomlinson, Allan. A distinguishing attack with a neural network, IEEE 13th International Conference on Data Mining Workshops, 2013. doi: 10.1109/ICDMW.2013.116

[Swapnaa,10] Swapnaa, Sammireddy; Dileepa, A.D.; Sekhara, C. Chandra; Kantb, Shri. Block cipher identification using support vector classification and regression, Journal of Discrete Mathematical Sciences and Cryptography v.13, n.4 , pp. 305–318, 2010. doi: 10.1080/09720529.2010.10698296.

[Torres,11] Torres, R.; Xexeo, J.; Souza, W.; Oliveira, G.; Linden, R. Identification of Keys and Cryptographic Algorithms Using Genetic Algorithm and Graph Theory, IEEE Latin America Transactions, v.9, n.2, pp.178-183, April, 2011. doi: 10.1109/TLA.2011.5765571

[Rukhin,10] Rukhin, Andrew; Soto, Juan; Nechvatal, James; Smid, Miles; Barker, Elaine; Leigh, Stefan; Levenson, Mark; Banks, David; Heckert, Alan; Dray, James; Vo, San. "A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications", National Institute of Standards and Technologies, NIST, Technology Administration, US Department of Commerce, Special Publication 800-22, Revision 1a, April, 2010.

[Xexeo,11] Xexeo, José; Souza, William; Torres, Renato; Oliveira, Glaucio; Linden, Ricardo. Identification of Keys and Cryptographic Algorithms Using Genetic Algorithm and Graph Theory, IEEE Latin America Transactions, v.9, n.2, 2011. doi: 10.1109/TLA.2011.5765571