Buletinul Ştiinţific al Universităţii "Politehnica" din Timişoara

Seria ELECTRONICĂ şi TELECOMUNICAŢII
TRANSACTIONS on ELECTRONICS and COMMUNICATIONS

Tom 54(68), Fascicola 2, 2009

Evaluating Noise Resistance and Speed for the New Generation of Symmetric-Key Encryption Algorithms

Ciprian Răcuciu[1], Dan Laurentiu Grecu[2],
Florin Medeleanu[3]

Abstract – In some appliances speed and noise resistance of the cryptographic algorithms are critical. This is the reason we analyzed these properties of some modern block ciphers, for the declared purpose of implementing the best algorithm in hardware modules (FPGA) in order to use it as high-speed encryption device. For in depth analysis only AES, Shacal and Trivium were processed. This article presents testing, evaluation procedures and results for the above analyzed block ciphers.

Keywords: symmetric-key encryption algorithm, Advanced Encryption Standard (AES).

I. INTRODUCTION

A cryptosystem is a system for encrypting a private message (plaintext) into a string (ciphertext), and decrypting the original plaintext. The ciphertext retains all the information of the private message, but appears random to statistical testing. At the core of every cryptosystem is an algorithm that combines a key and the plaintext to achieve the ciphertext as an output. Cryptanalysis is the term given to the branch of cryptography concerned with mathematical analysis of the security of a cryptosystem. Performance analysis of cryptosystems has become more important to the cryptographic community recently, because of the increased need for secure communication in a much wider set of environments, especially in the emerging area of e-commerce, smart-cards, PDAs, and mobile and wireless communications.

II. Background

Most modern cryptosystems are built around symmetric block ciphers. A symmetric block cipher is a specific form of cipher, where plaintext is encrypted a block at a time and is decrypted using the same key. The difference between symmetric ciphers and asymmetric ciphers in terms of their operation is that asymmetric ciphers do not require both parties knowing the same secret key. Asymmetric encryption is also much slower than symmetric encryption. The predominant method of encryption is to use an asymmetric cipher to encrypt the key for a symmetric cipher that is then used in communicating a secret message.

The Data Encryption Standard (DES) was issued as a Federal Information Processing Standard in July 1977 and since 1999, DES is only used in legacy systems because it has become insecure, due to developments in both hardware and mathematical cryptanalysis. DES belongs to a group of symmetric bloc ciphers known as Feistel networks. A Feistel network is traditionally built such that the text being encrypted is split into two halves. A function is applied to one half with the introduction of the key, and then the Boolean exclusive-or (XOR) operator is applied to the result of the function and the other half. The two halves are then swapped. Many modern ciphers are based on a generalization of this structure, as proposed by Schneier in 1996. Feistel structure is desirable in ciphers since encryption and decryption are structurally identical. In 1996, 44% of 1393 encryption products identified worldwide implemented the DES. In early 1998, DES was broken in 56 hours (Landau 2000), confirming the obsolescence of DES, the ciphertext was decrypted by finding a 56-bit key using dedicated hardware.

In 1998, the National Institute of Standards and Technology (NIST) issued a challenge to the cryptographic community, proposing a competition to replace the DES. The Advanced Encryption Standard (AES) was the outcome of this competition. Fifteen algorithms were submitted to the first round of competition for the AES. The eventual winner of the competition was a cryptosystem developed by Vincent Rijmen and Joan Daemen. It was named Rijndael and the general structure of the cipher that forms the basis for the cryptosystem is a substitution linear transform network (SLTN). This is a form of a substitution permutation network (SPN) involving a layer of linear transform in each round.

[1] Military Technical Academy, Bucharest, e-mail ciprian.racuciu@gmail.com
[2] Ministry of National Defense, Bucharest, email danlaurentiugrecu@gmail.com
[3] Ministry of National Defense, Bucharest, email florinmed@yahoo.com

Interest shifted away from performance in 2002 because of the controversy surrounding the new information regarding possible attacks on two of the finalists of the AES competition. There has always been a continual cycle of new cryptanalysis techniques being developed, causing designers of encryption algorithms to formulate more secure algorithms. A consequence is that there is always a need for performance measurement of state-of-the-art encryption algorithms that push the boundary of security will inevitably require greater time and space complexity. After AES competition there were many other competitions for signatures, integrity and encryption (CRYPTREC, NESSIE, E-STREAM etc.).

III. TEST OF NOISE RESISTANCE

During of data transmission, any canal of transmission is introducing transmission errors. The influence of these errors depends on the algorithm.
In order to establish which algorithm is the most appropriate for operation in noisy environment, some tests were performed for different values of BER (Bit Error Rate). The graphical result shows the influence of the noise upon the signal. A bitmap image was used as a plaintext, the resulting cipher text was altered for different values of BER, then the result was decrypted and differences between the original file and the decrypted file were counted. We tested Trivium, Shacal and AES in ECB, CBC and OFB mode as follows.
The results are shown in the tables below.

Table 1 - Noise resistance of Shacal ECB

BER	Wrong bytes	Wrong blocks
10^{-3}	709318/3175578	39569/99238
10^{-4}	78120/3175578	4863/99238
10^{-5}	7945/3175578	500/99238
10^{-6}	802/3175578	52/99238

Table 2 - Noise resistance of Shacal CBC

BER	Wrong bytes	Wrong blocks
10^{-3}	734098/3175578	49249/99238
10^{-4}	83526/3175578	6694/99238
10^{-5}	9246 /3175578	758/99238
10^{-6}	665/3175578	57/99238

Table 3 - Noise resistance of AES ECB

BER	Wrong bytes	Wrong blocks
10^{-3}	380171/3175578	44858/198475
10^{-4}	40554/3175578	5051/198475
10^{-5}	3797 /3175578	474/198475
10^{-6}	460/3175578	58/198475

Table 4 - Noise resistance of AES CBC

BER	Wrong bytes	Wrong blocks
10^{-3}	402179/3175578	51946/198475
10^{-4}	43683/3175578	6062/198475
10^{-5}	3924/3175578	550/198475
10^{-6}	559/3175578	77/198475

Table 5 - Noise resistance of AES OFB

BER	Wrong bytes
10^{-3}	25533/3175578
10^{-4}	2593/3175578
10^{-5}	232/3175578
10^{-6}	29/3175578

Table 6 - Noise resistance of Trivium

BER	Wrong bytes
10^{-3}	25374/3175578
10^{-4}	2541/3175578
10^{-5}	257/3175578
10^{-6}	19/3175578

Table 7 – Algorithm placement

Algorithm	Place
Trivium	1
AES OFB	2
AES ECB	3
AES CBC	4
Shacal ECB	5
Shacal CBC	6

IV. SPEED EVALUATION

In this paper we adopted the model of speed evaluation used by the NESSIE project.
This project tested the performance of 285 implementations of 138 different variants of primitives.
The tests were performed on 11 different kinds of platforms (on over 20 computers), on various operating systems and with various compilers. On some processors (e.g, Pentiums) there were made the tests on two operating systems with 4 compilers, and even several different versions of some compilers, in order to achieve the best results. In total, the performance tests ran several thousands of computer hours.
NESSIE tested symmetric candidates along with standard primitives and many 'non-standard' primitives.
The tool measures the time for key setup, encryption, decryption, IV setup, and hash and MAC initialization and finalization.
The tool checks the correctness of all codes by comparing the encryption results to the supplied test vectors. For encryption, e.g., the time is measured in the following way (decryption, key setup, ... analogously):
– First, the random plaintexts are encrypted for about one second. Based on the number of plaintexts encrypted in one second, it is estimated how many encryptions are expected to run in 10 seconds.
– Then, the estimated numbers of encryptions are run and their run time is measured.
The actual measurement is executed with many different keys for many different encryption/ decryption blocks. It is calculated the encryption, decryption, hash, MAC time in units of cycles/byte and the key setup, IV setup time and hash and MAC

initialization and finalization in cycles/invocation. The results are compared on various machines.
The results show very high consistency between different machines of the same type, especially between various PIIIs (at different speeds and with different memory sizes).
For the measurement of speed, all ciphers were compiled with all the available compilers, with various optimization options (as adequate for the machine and compiler), and selected the best speed that resulted from all these options.
In many cases, higher optimizations (such as -O3) resulted in poorer speeds than lower optimizations (such as -O1), and in many cases optimizations targeted to older processors (such as optimization targeted for 386 when running on PIII) gave better results than optimizations targeted to the newer ones (such as Pentium or Pentium-pro).
For this reason, on most machines, the measurements consisted of more than a dozen compilations with different optimization options and target machines, to ensure that the best code the compiler can generate was not missed. In the case of PIII with Linux, the measurement was performed under three different versions of the gcc compiler, with over 40 different optimization options for the newer version.
It was also ensured that the compiled code is correct by regenerating the test vectors in each run with each compilation option. In those rare cases where some compilation option generated wrong code on some machine, the speed results of the runs with the wrong results were ignored.
It was also ensured that the main test program was compiled with the same optimization option in all cases, although the code of the primitives was compiled with different options, in order to make the overhead of the test program as fixed as possible.
In order to measure this overhead, the speed of dummy ciphers (that do nothing) was measured and verified that their computation time is negligible.
It should be noted that all codes (of each family of primitives) use the same API (which, among other things, ensures that the keys are set up into structures that can later be passed as parameters to the encryption (decryption, etc.) function, and that no global or static variables depending on the key, state, ... are used), and thus, the overhead of all codes of the same type and block size is expected to be similar.
It can be seen from the results that the codes for the primitives are quite optimized.
This is the result of several rounds of optimizations of the submitted codes by several people in the NESSIE project. For about 50% of the ciphers, the codes were even faster than the submitters claim, and for several others ciphers, the results are only a few cycles slower than claimed.
In most cases, the order of the primitives by decreasing speed is similar on all machines. Exceptions are primitives that are optimized for 64-bit machines, which become the fastest on Alpha, although they are not so on other machines.
Two examples are RC6 with 256-bit blocks (using 64-bit multiplications), which, on Alpha, is even twice faster than the standard RC6 with 128-bit blocks, although it is twice slower on all other machines, and Tiger, which, on Alpha, becomes even faster than MD4, although it only has a medium speed on all other machines.
Note that the same implementations of block ciphers and stream ciphers were also subjected to the NESSIE statistical tests, and all of them passed these tests.
It was also measured the amount of memory required for the various implementations, and verified that the speeds reported can be reached with a reasonable amount of memory.
There were not distinguished cases where the key setup can be faster for encryption-only or decryption-only applications. In all cases, the time required for a full key setup was measured. In the cases of Rijndael, and several others, the time required to setup encryption-only keys may be significantly faster than the full time of the key setup.
In addition, claimed performance for some algorithms, as the developers present it, is described below.

Table 8 - Claimed performance of AES

Processor	Block size	Cycles/byte	MBytes/s
Pentium II/III	128	28,6	53,3

Table 9 - Claimed performance of Shacal-1

Processor	Block size	Encr/Decr (cycl./byte)	Key setup (cycl./byte)
Pentium II	160	140/116,5	3200
Pentium III	160	124/116	2280

Table 10 - Claimed performance of Shacal-2

Processor	Block size	Encr/Decr (cycl./byte)	Key setup (cycl./byte)
Pentium II/III	256	112.5/115	2800

V. COMPARING EXECUTION TIME

In our project, we performed comparing for a C implementation running on the same PC, using as plaintext a bitmap file which size is 3175578 bytes. The test should be seen as a relative one, not as an absolute one, with the solely purpose of placing the analyzed algorithms.
Shacal is faster than Rijndael, due to the fact it contains mainly simple operations. For this, an FPGA implementation of Shacal should encrypt faster than Rijndael. The present implementation of Rijndael doesn't optimize speed, the multiplication in Galois fields taking long time. The implementation should be faster if the results of multiplications in Galois fields would be pre-calculated.

Table 8 – Execution time

Algorithm	Key expansion time	Encryption time
SHACAL2 ECB	19us	1s 183ms 108us
SHACAL2 CBC	19us	1s 192ms 842us
RIJNDAEL ECB	54us	21s 510ms 94us
RIJNDAEL CBC	54us	22s 120ms 532us

V. CONCLUSIONS AND FURTHER WORK

- The performed analysis concerning noise resistance and speed execution of cipher algorithms within this article contains qualitative data that can help for cryptographic systems design. The impact of this analysis refers especially to emission/reception buffers dimension and to hardware structure choice of cryptographic systems (or encryption equipments). In the same time this information can help to suitable dimension of strategies addressing the quality of service QoS.
- Hardware implementation for block ciphers is possible using different type of hardware: FPGA, ASIC etc. Devices such as Field Programmable Gate Arrays (FPGA-s) are the highly attractive option for hardware implementations of encryption algorithms. For this reason, FPGA technology was widely used within NESSIE project for hardware implementation of encryption algorithms. Technological level of NESSIE project is longtime overcome, however general conclusions can be translated to the current technology. However, for current projects, FPGA implementations of a symmetric key encryption algorithm within a specific target technology must be considered.
- The AES is still considered present encryption standard, even though recent researches proved that the time complexity of attacks for key recovering dropped to an alarming 2^{112}. The expected reaction of NIST is to announce a competition for a new standard or to raise the number of rounds of the current AES.

REFERENCES

[1] Data Encryption Standard, FIPS 46-1;
[2] Request for comments on the finalist (round 2) candidate algorithms for the advanced encryption standard (AES), Federal Register;
[3] Feistel. H, “Cryptography and computer privacy”, Scientific American 228 (5), 15-23;
[4] Schneier, B., Kelsey, J., “Unbalanced Feistel networks and block cipher design”, Fast Software Encryption, Third International Workshop Proceedings, pp. 121-144;
[5] Denning, D.E., “Encryption policy and market trends”, http://www.cs.georgetown.edu/denning/crypto/trends.html;
[6] Curtois, N.T., Pieprzyk, J., “Cryptanalysis of block ciphers with overdefined systems of equations”;
[7] Daemen, J., Rijmen, V, “Aes proposal: Rijndael”;
[8] S. Park, S.H. Sung, S. Lee, J. Lim, “Improving the upper bound on the maximum differential and the maximum linear hull probability for SPN structures and AES”, Fast Software Encryption, FSE 2003, in Lecture Notes in Computer Science, Vol. 2887, Springer, Berlin, 2003, pp. 247-260;
[9] F. Sano, K. Ohkuma, H. Shimizu, S. Kawamura, “On the security of nested SPN cipher against the differential and linear cryptanalysis”, IEICE Trans. Fund. Elec., Commun. and Comput. Sci. E86-A (1) (2003) pp. 37-46.
[10] S. Vaudenay, “On the security of CS-Cipher”, Fast Software Encryption FSE'99, in: Lecture Notes in Comput. Sci., Vol. 1636, Springer, Berlin, 1999, pp. 260-274;
[11] L. Keliher, H. Meijer, S. Tavares, “Improving the upper bound on the maximum average linear hull probability for Rijndael”, Selected Areas in Cryptography SAC 2001, in: Lecture Notes in Comput. Sci., Vol. 2259, Springer, Berlin, 2001, pp. 112-128.
[12] P. Serf, “The degrees of completeness, of avalanche effect, and of strict avalanche criterion for Mars, RC6, Rijndael, Serpent, and Twofish with reduced number of rounds.” Public report, NESSIE, 2000 [p. 73]
[13] L. Keliher and J. Sui, “Exact maximum expected differential and linear probability for 2-round Advanced Encryption Standard (AES)”;
[14] B. Preneel, A. Biryukov, C. De Canniere, “Final report of project New European Schemes for Signatures, Integrity, and Encryption (NESSIE)”, Springer-Verlag Berlin, 2004;
[15] Information Technology Promotion Agency of Japan, Telecommunications Advancement Organization of Japan, “Cryptrec report 2001&2002”;
[16] A. Klapper, M. Goresky, “Feedback shift registers, 2-adic span, and combiners with memory”, Journal of Cryptology, vol. 10, no. 2, pp. 111–147, 1997. [p. 73, 154];
[17] A. Bibliowicz, P. Cohen, and E. Biham, “A system for assisting analysis of some block ciphers.” Public report, NESSIE, 2002 [p. 74];