Attacking Weak Symmetric Ciphers

HW1 - CNS Sapienza

Pietro Borrello

02 Nov 17

1 Introduction

Due to the continuous increasing of computational power, every encryption algorithm will eventually become obsolete. Any algorithm for which a bruteforce attack is computationally feasible, is considered weak, and must not be used. In this article we present an analysis on the bruteforce attack against such an algorithm as DES, for which we provide an implementation and performance results.

DES, first published in 1975 by IBM, provide symmetric encryption with a 56bits key. It has been considered obsolete since 1998, thanks to the EFF DES cracker [1] that managed to broke a DES encrypted message in 56 hours exploiting the limited key space given by such a small key. Apart form bruteforce more advanced techniques exist, as Differential and Linear Cryptanalysis on DES [2] that can break the cipher in less than the complexity of the exhaustive search in the key space. Such attacks are not considered in this article, that has the aim to show that DES is breakable only with a dumb bruteforce attack. Thus providing the thesis that using DES should be foolish.

2 Resources Used

To test the implementation a 2 GHz Intel Core i5 (dual core, with 4 virtual cores thanks to hyper-threading) with 16 GB of RAM had been used.

The choice of the technology to implement the cracker was driven by the will to use *libgcrypt*. We consider this library the state of the art in cryptographic libraries, with continuous updates and bug fixes on it, giving us the confidence that all the operation would be implemented in the best possible way. *Libgcrypt* is a C library developed as part of GnuPG, in the GNU Project, approved in Federal Information Processing Standard Publication 140-2 by NIST.

3 Implementation

As previously said, we based our implementation on *libgcrypt* library, providing as a proof of concept a C program, that has the aim to crack a key of a given complexity, from the knowledge of a pair of plaintext and ciphertext. A DES key is composed by 56bits, split in 8 groups of 7bits, and for each group of bits, a parity bits is added, resulting in 8 groups of 8bits, maintained in a 64bit vector.

To make the approach scalable, we limited the key space to test the implementation on machines that are not so powerful. So we defined the complexity of the key for the cracker as the total number of unfixed bytes in the key vector. Thus $8 - COMPLEXITY$ provides the number of the bytes in the vector we fixed to zero, for testing purpose. For example a key of complexity 5 is:

0	0	0	x	x	x	x	x

and only the unknown bytes set to x have to be cracked.

Therefore a $COMPLEXITY$ set to 8 means a full DES crack. All the unfixed bytes are randomized, to provide a performance estimation on the average case and not only in the worst case, that would be obtained fixing the bytes at $0xff$.

The key enumeration in the bruteforce as to be made carefully: to be noticed is that not all the libraries (and neither *libgcrypt*) check parity bits, simplifying the code of the key enumeration. So a careless designer could think that the key enumeration could be simply done increasing a long integer index in a 64bit environment. This naive implementation would increase key space from 2^{56} to 2^{64}. Our implementation, considered not naive, is a simple reimplementation of carriage addition in algebra considering only 7 bits out of 8. The snippet of the code follows:

References

[1] S. Landau, *Standing the test of time: The data encryption standard*, Notices of AMS, 2000.

[2] Eli Biham, Adi Shamir, *Differential Cryptanalysis of the Full 16-Round DES*, Springer, 1993.