Tagada
Tool for Automatically Generation of Abstraction-based Differential Attacks

Luc Libralesso[1], François Delobel[1], Ana Margarita Rodriguez Cordero[2***†], Pascal Lafourcade[1], Christine Solnon[3]

[1] LIMOS, CNRS UMR 6158, University Clermont Auvergne, France
[2] Université de Lorraine, CNRS, Inria, LORIA, Nancy, France
[3] INSA Lyon, CITI, INRIA CHROMA, F-69621 Villeurbanne, France

Proving security of block ciphers is crucial while building them. A very powerful attack used against them is differential attack. Given a block cipher E, a plaintext P and an unknown key K, differential attacks study the propagation of input differences throughout the cipher:

$$\nabla = E_K(P) \oplus E_K(P \oplus \varDelta).$$

A general approach applied to word-based block ciphers is the search of differential characteristics through the use of truncated differential characteristics. This is done generally in two steps, a first step: the Abstraction Step and a second step usually named Enumeration Step.

The abstraction is done by assigning a binary variable to each word of the block (*i.e.* nibble, bytes,...), having value one when there is a difference in the corresponding position (*i.e.* an active word) and zero in the opposite case. The abstraction is used to search for the truncated differential characteristic that might lead to a differential characteristic minimizing the number of active S-boxes. The process is divided into two parts, Step1-opt searching for the minimum number of active S-boxes and Step1-enum searching for all the truncated differential characteristics.

The enumeration step aims to recover actual differential characteristics from the truncated ones, finding the concrete value for each abstracted word.

This security study demands a lot of work for each cipher. It requires the application of dedicated solutions and the knowledge of different solvers and programming languages. It is a very time-consuming and prone-error process. However, the basis and means for this analysis are often common from one cipher to the other and ciphers usually have the same kind of underlying constructions, a fact that leads to the idea that the process could be automatized.

TAGADA aims to tackle this issue and to provide a tool that can analyze any cipher automatically, removing the burden of creating a different solution for each cipher. As an input, TAGADA receives a description of the cipher through black-box operators and Directed Acyclic Graphs (DAGs). From these inputs it is able to generate constraints and MiniZinc optimization models.

Ciphers are described in TAGADA through a dedicated language. In this language, the cipher description is given by the declaration of each of its components by means of operators. Any type of operator can be implemented in the tool. They are treated as black boxes, generating from their inputs and outputs constraints modelling how the propagation of differences occurs at each point. In the given case that they do not affect the difference propagation, they are removed with a shaving process.

Attacks are performed in TAGADA under two different instances: the single key and related key scenarios. We will talk about how to implement these attacks in order to generate them automatically, working with several optimization and enumeration problems.

For the Abstraction Step of the attack, we are able to obtain competitive results, having thus far abstraction levels of the quality of state-of-the-art approaches for Midori, AES and CRAFT.

Key words: Symmetric Cryptography, Differential Cryptanalysis, Truncated differentials, Constraint Programming, SAT, ILP.

*** Internship performed at LIMOS lab, under the Decrypt ANR project.
† Speaker: Ana Margarita Rodríguez Cordero - PhD student