1. What does this source code do?

This code finds all differential characteristics of 13 rounds SKINNY-64 in TK-1 model under the assumption of having a single active bit in input and output states. In this code we have only considered the differential models as $\Delta(S_1^1), \Delta(tk_1^1), \Delta(ouput)$ for SKINNY in TK-1 model.
At first, we construct MILP model for SKINNY-64 in TK-1 model based on [1,2]. For modeling the SKINNY S-box we use algorithm 1 in [1] and SAGE software [4]. After modeling MILP of SKINNY, the main strategy in this code is the same of approaches in [2,3]. In fact, at first we set all values of $Δ(S_1^1), Δ(tk_1^1), Δ(output)_$ to "0". Then, we consider the first bit of $Δ(S_1^1)$, the first bit of $Δ(tk_1^1)$, and the first bit of $Δ(output)$ as "1" and solve the model. If the model was infeasible, we have an RK_ID characteristic and if it is feasible, this choice of input/output differentials is not RK_ID. We continue this approach for all cases of $\Delta(S_1^1), \Delta(tk_1^1), \Delta(ouput)$ with a single bit difference.

1.1 Notations:

The following notations are used in the source code (also see Figure 1):
For i=0,…15:
Si [r] [0] [j]: represents difference of the j-th bit (j = 0,…,4) of the i-th cell of S-box input in the r-th round.
Si [r] [1] [j]: represents difference of the j-th bit (j = 0,…,4) of the i-th cell of S-box output in the r-th round.
Si [r] [2] [j]: represents difference of the j-th bit (j = 0,…,4) of the the i-th cell of the state after adding tweakeys in the r-th round.
Si [r] [4] [j]: represents difference of the j-th bit (j = 0,…,4) of the i-th cell of Mix column input in the r-th round.
Si [r] [5] [j]: represents difference of the j-th bit (j = 0,…,4) of the i-th cell of Mix column output in the r-th round.

TKi [r] [0] [j]: represents difference of the j-th bit (j = 0,…,4) of the i-th cell of TK input in the r-th round.

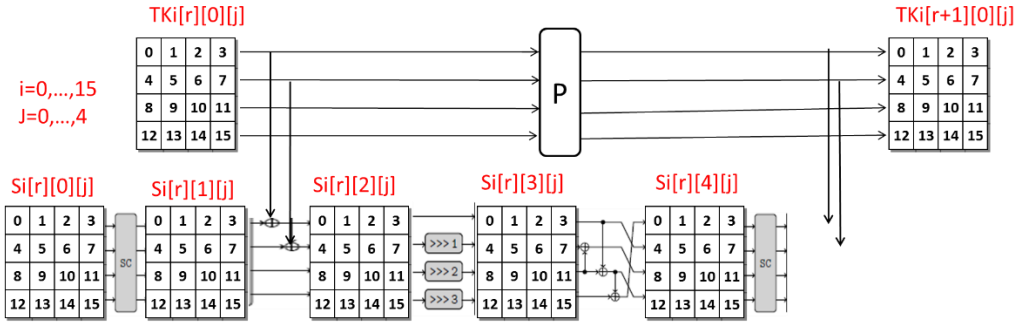

Figure 1. Notations of the r-th round of SKINNY in TK-1 model that are used in the source code

In the source code, we define
"inOput_to_zero" function to set all input/output differentials to "0",
"Sbox" function to model the differential patterns of the S-boxes SKINNY. We have obtained these patterns by using Algorithm 1 in [1] and SAGE software [4].
"TK01" function to model tweakey structure for SKINNY in TK-1 model,
"XORR" function to model the XOR operation in MILP.

2. How to get up and running with Gurobi in Microsoft Visual Studio

The following steps demonstrate how to achieve this for a mixed integer programming problem:

2.1. Install & Licensee Gurobi

Run the Gurobi installer file (get it from http://www.gurobi.com/). At the time of writing our code, we installed 64-bit version of Gurobi 7.5.2.

2.2. Create a new Visual Studio project

We are using the Visual Studio 2017, though Gurobi should be suitable for version 2015 as well.
To keep things simple, open Visual Studio C++ project.

3. Set the additional include directories

Before writing any code, set the Visual Studio project dependencies, starting with the additional include files it will need.
Right-click on your project folder and select properties. In the displayed dialog box, select C/C++ > General > Additional Include Directories. Set this value according to how you installed Gurobi. In our installation, the include files are located in the folder 'C:\gurobi752\win64\include'.

4. Set the library dependencies

Right-click your project folder and select properties. In the displayed dialog box, select Linker > General > Additional Library Directories. Set the directory to the location of where your Gurobi library files are located. In our installation this location is C:\gurobi752\win64\lib

In the same dialog box, select the Linker > Input tab and set the variables needed for the Additional Dependencies. For Visual Studio 2017 and Gurobi version 7.5.2, the additional dependencies needed are :
gurobi_c++mdd2017.lib
gurobi75.lib
Once these are set click OK.

References:

[1] Sun, Siwei, et al. "Towards finding the best characteristics of some bit-oriented block ciphers and automatic enumeration of (related-key) differential and linear characteristics with predefined properties." Cryptology ePrint Archive, Report 747 (2014): 2014.
[2] Cui, Tingting, et al. "New Automatic Search Tool for Impossible Differentials and Zero-Correlation Linear Approximations." IACR Cryptology ePrint Archive 2016 (2016): 689.
[3] Sasaki, Yu, and Yosuke Todo. "New impossible differential search tool from design and cryptanalysis aspects." Annual International Conference on the Theory and Applications of Cryptographic Techniques. Springer, Cham, 2017.
[4] Stein, William. "Sage mathematics software." http://www.sagemath.org/ (2007).
[5] Optimization, Gurobi. "Inc.,"Gurobi optimizer reference manual," 2015." URL: http://www.gurobi.com (2014).