BigProject in Cybersecurity

Cryptanalysis of ForkAES-*-2-2

Tutor: Arnab Roy *Student: Luca Campa*

1 Attack on ForkAES-*-2-2 with Reflection Trails

The attack can work for an arbitrary number of rounds before the state is forked. Our analysis and diagrams refer to 5 rounds before the fork and 2 rounds after that, then the keys are numbered from 0 to 9.
The analysis will make use of the diagram in Figure 1 because only the two rounds after the fork will be considered[1].

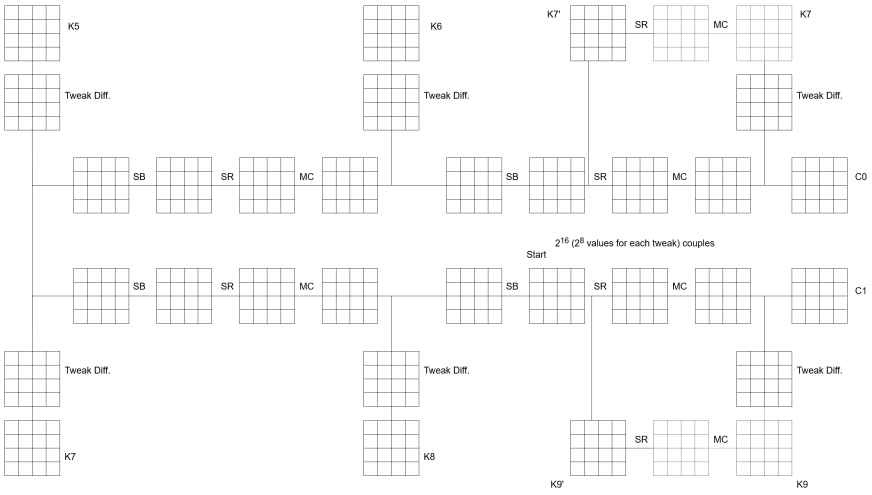

Figure 1: Considered part of the cipher

Reflection Trails The analysis will make use of a modified version of the Reflection property which considers the differences between the states of two different encryptions. In particular, we are searching for a differential characteristic which makes possible to find one of the keys involved in the tag-reconstruction operation.

Proposition: Differential Reflection Trail If there exists a differential F_r for the r-round transformation that propagates a difference ΔI to ΔO with probability p, there exists a differential for the 2r-round transformation $(F^{-1})^r \circ F^r$ that propagates a difference ΔI to ΔO with probability at least p^2. This property holds for any choice of round keys and constants in the 2r rounds[2].
We can apply Reflection trails technique on the tag-reconstruction procedure because it is a back and forward computation of the same round function with different round-keys and constants. The addition of constants and round-keys does not interfere with the property we are going to use.

[1] Notice that the values indicated with the apex in the diagrams are shown with a ˜ in the text
[2] Cryptanalysis of ForkAES, page 6

The characteristic Thanks to the mix column operation, a single active byte propagates to 4 in the next round and to 16 in the second one ($1 \xrightarrow{F} 4 \xrightarrow{F} 16 \xrightarrow{F} 16$. Tweak injection helps us to get better trails. Notice that, if we decide a specific tweak difference, we can find a couple of states whos difference from the fork is $1 \xrightarrow{F} 0 \xrightarrow{F} 1 \xrightarrow{F} 4$. That means that whilst the common diffusion is reduced, the probability of guessing the key increases.

The reflection trail From the above differential characteristic, we can build a reflection trail which enables the attacker to break the cipher efficiently. The trail is: $4 \xrightarrow{F^{-1}} 1 \xrightarrow{F^{-1}} 0 \xrightarrow{F^{-1}} 1 \xrightarrow{F} 0 \xrightarrow{F} 1 \xrightarrow{F} 4$.

Attack The probability that the one byte difference in $\tilde{C}_1$ becomes equal to the tweak difference after the inverse sub bytes operation in order to make it disappear is 2^8. This means that for each byte of the key we can obtain a very little number of possibilities (1-2) by only considering $2^8 \times 2^8$ couples of ciphertexts C_0.
Such trail does not work with 3 or 4 rounds after the fork because it will imply that the states at the fifth round should be equal, which is slightly impossible.

Attack Procedure for one key byte With the following procedure we are going to find one byte of $\tilde{K}_7$.

1) Choose tweaks T_0 and T_1 with the chosen fixed difference ΔT. This means that you will end up with 2^8 couples (T_0, T_1).

2) For each tweaks couple:

- For each tweak T in the couple:
  - take the 2^8 distinct values of $\tilde{C}_1$ for the same byte number in which the tweak difference is considered. Meaning that, if the tweak difference is at byte 0, we will take all the possible values of the byte 0 of $\tilde{C}_1$. Fix the other 15 bytes to constant values and compute the corresponding C_1
  - You will obtain 2^8 values of C_1. Query each of those values to the tag-reconstruction procedure in order to obtain the corresponding C_0 values. For each of those 2^8 C_0 values, compute the corresponding $\tilde{C}_0$.
- We obtain 2^8 values computed with T_0 and 2^8 values computed with T_1.
- From the 2^{16} couples between different tweaks, pick the one with 15 inactive bytes in the byte positions different from the one of the tweak difference. If such couple is not found, skip to the next tweak couple, otherwise pick such couple and go to step 3.

3) We expect at least one right pair. Initialize a counter for each possible byte value of the key we are trying to find. For each possible byte value of $\tilde{K}_7$:

- For each value in the couple:
  - add that key byte, add the corresponding tweak (T_0 for the first value in the couple and T_1 for the second one), apply inverse mix-column and inverse shift-row.

– If there is no difference between the two computed states increase the counter of the tried key byte value.

4) Consider only the values whos counter value is equal to the maximum. We expect at most 2 values with a probability of the 95,4% (computed empirically).

Full Attack Iterate the procedure above for each key byte (the figures in Appedix show the trails for the key bytes of $\tilde{K_7}$ column 0). After having computed the possibilities for each key byte, compute all the possible keys $\tilde{K_7}$ and test them progressively. Once the possibilities are reduced to one key candidate, compute the corresponding K_7. From the computed K_7, generate all the other round keys by applying the key-scheduling and the inverse-key-scheduling algorithm.

Possible Countermeasures What it makes the attack easier to apply is the injection of the tweak, which is never modified during the process. That characteristic helps us to reduced the complexity of the differential trail. Then, a possible countermeasure could be to modify the tweak during the process in such a way that the tweaks applied in the two branches are always different. In particular, a permutation of the bytes could be applied. In our example, 5+2 rounds ForkAES, we would need to generate tweaks from T_0 to T_9 as done with the round keys. The application of the round tweaks should be done by adding the round tweak i to the round i state and so on.
A possible simple implementation can be found on the repository. In particular, the file `utilities.c` contains a procedure called `TweakExpansion` which permutes the value within the initial tweak.

2 Appendix

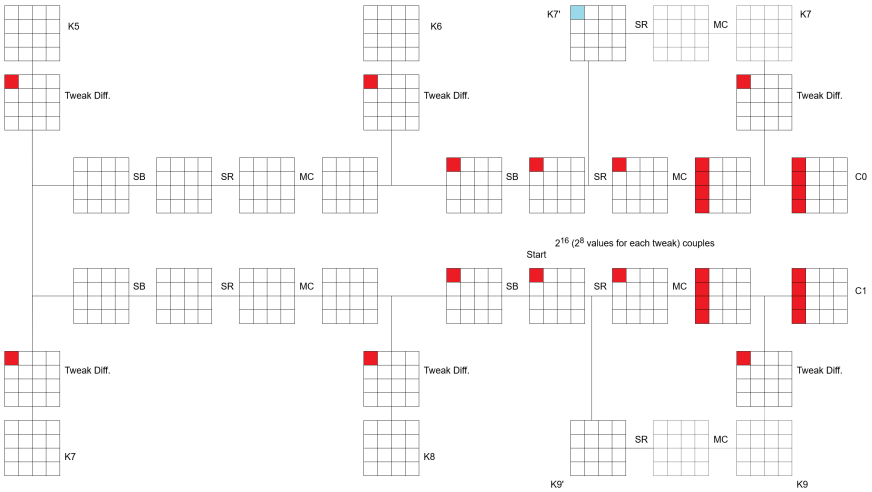

Figure 2: Differential Reflection Trail: Byte 0

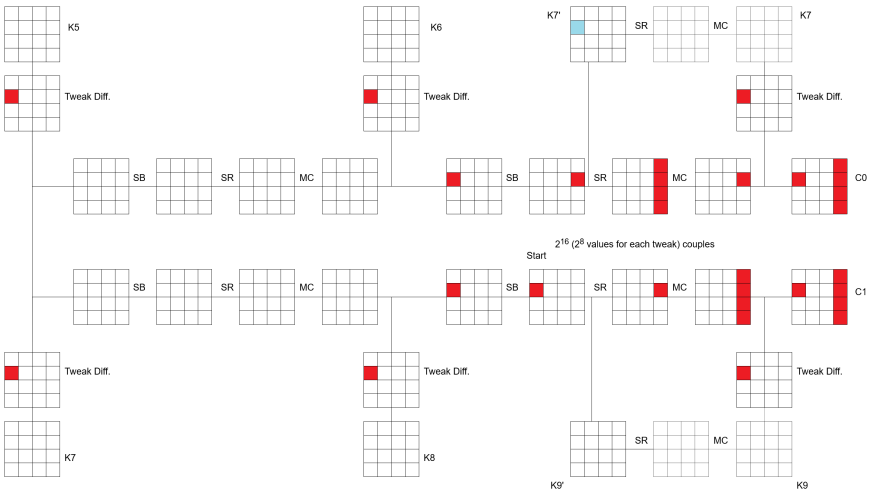

Figure 3: Differential Reflection Trail: Byte 1

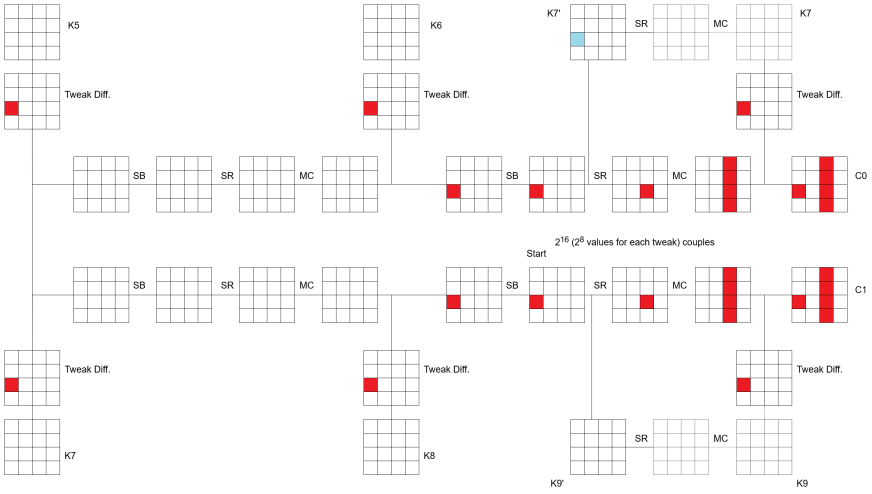

Figure 4: Differential Reflection Trail: Byte 2

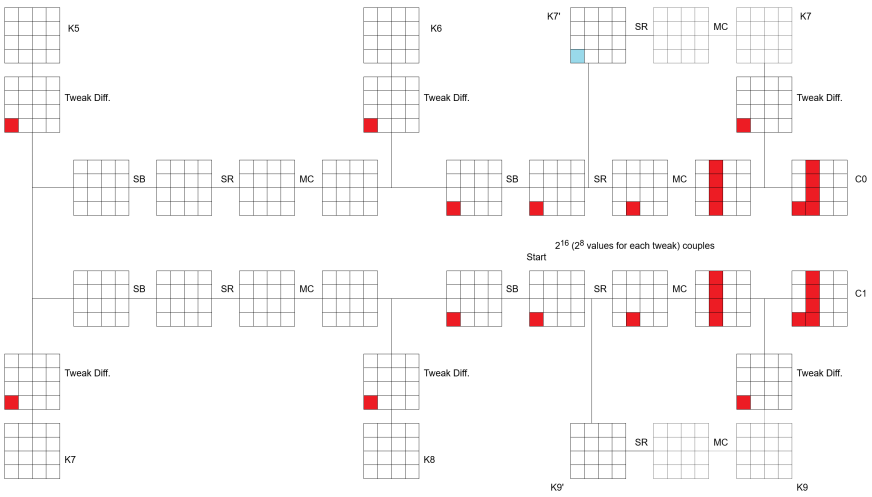

Figure 5: Differential Reflection Trail: Byte 3