UNIVERSITÀ DEGLI STUDI DI PADOVA

DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE

# Information Security 2023-2024

## Laboratory session 1

## Implementation and linear cryptanalysis of a simplified AES-like cipher

| **Group Acronym** | ZUC | |
|---|---|---|
| **Last Name** | **First Name** | **Badge Number** |
| Ragusa | Simone | 2104296 |
| Caripoti | Eugenio | 2079454 |
| Gusella | Michele | 2122861 |
| Nemanja | Lalic | 2085346 |
| Pozzo | Nicola | 2125382 |

# 1 Tasks 1, 2, 5 & 7

## 1.1 Overall implementation choices

We choose to implement all the instantiations of the cipher, i.e., linear, nearly linear, and non-linear, using SageMath, a computer algebra system (CAS) based on Python. SageMath automatically handles the computations in modulo ($p = 11$, here), so this choice allows us to focus on the real challenges of this project.

The cipher implementation follows a generic approach: we define wrapper functions for all those components that differ depending on the instantiation (i.e., the substitution function and the subkey generation), and then we make the corresponding implementations for each instantiation. The subkey addition function, the linear transformation, and the transposition function are the same for all the instantiations.

Each function, except the transposition one, takes an additional input parameter for specifying the computation of its inverse. Since the transposition function is the inverse of itself, it does not have this parameter.

Finally, we implement two other wrapper functions, one for the encryption operation and one for the decryption operation. These functions use the component wrappers to encrypt a given plaintext (or decrypt a given ciphertext) by going through all the required rounds.

With all that in place, the actual implementation of the cipher is straightforward, and we don't encounter any real difficulty in realizing any of the different instantiations.

# 2 Task 3

## 2.1 Description

Assuming that $u$ is the plaintext, $k$ is the key, and $x$ is the ciphertext, the linear instantiation of the cipher can be reduced to the following linear equation, where $A$ and $B$ are matrices of appropriate dimensions.

$$x = Ak + Bu \mod p \tag{1}$$

Following the procedure outlined in Appendix 1 of the project instructions, we find matrix A by setting the plaintext to the all-zero vector and the key progressively to the successive standard orthonormal basis vector. Similarly, we find matrix B, switching the roles of key and plaintext.

## 2.2 Results

The resulting matrices that constitute the linear relationship representing the linear instantiation of the cipher are as follows.

$$A = \begin{bmatrix} 9 & 0 & 1 & 6 & 0 & 0 & 1 & 10 \\ 0 & 8 & 6 & 2 & 2 & 9 & 0 & 0 \\ 0 & 6 & 0 & 8 & 3 & 10 & 0 & 0 \\ 6 & 0 & 0 & 8 & 0 & 1 & 6 & 6 \\ 2 & 0 & 1 & 10 & 0 & 0 & 1 & 3 \\ 0 & 1 & 8 & 4 & 9 & 6 & 0 & 0 \\ 0 & 10 & 0 & 5 & 7 & 6 & 0 & 0 \\ 3 & 0 & 0 & 1 & 0 & 1 & 4 & 8 \end{bmatrix} \quad B = \begin{bmatrix} 6 & 0 & 0 & 3 & 3 & 0 & 0 & 0 \\ 0 & 6 & 3 & 0 & 0 & 3 & 0 & 0 \\ 0 & 3 & 6 & 0 & 0 & 0 & 3 & 0 \\ 3 & 0 & 0 & 6 & 0 & 0 & 0 & 3 \\ 5 & 0 & 0 & 0 & 4 & 0 & 0 & 8 \\ 0 & 5 & 0 & 0 & 0 & 4 & 8 & 0 \\ 0 & 0 & 5 & 0 & 0 & 8 & 4 & 0 \\ 0 & 0 & 0 & 5 & 8 & 0 & 0 & 4 \end{bmatrix} \tag{2}$$

# 3 Task 4

## 3.1 Description

It is easy to carry out a known-plaintext attack on the linear instantiation of the cipher. Indeed, once a plaintext-ciphertext pair is known, we can easily find the key by inverting the linear equation (1) as follows.

$$k = A^{-1}(x - Bu) \mod p \tag{3}$$

Since we have at our disposal five plaintext-ciphertext pairs, and we know that they are all encrypted with the same key, we can also check that what we're doing is correct by verifying that all the resulting keys are equal.

In conclusion, considering the matrices in (2), we compute the key for each of the five plaintext-ciphertext pairs as in (3).

## 3.2 Results

For each of the five plaintext-ciphertext pairs, the key recovered using a known-plaintext attack on the linear instantiation of the cipher is:

$$\hat{k} = [3, 2, 6, 5, 8, 9, 8, 10].$$

# 4 Task 6

## 4.1 Description

To find a linear approximation of the nearly linear instantiation of the cipher and subsequently perform a known-plaintext attack, we can take two approaches: guess one good linear approximation by directly looking at the components of the cipher or performing a full-blown linear cryptanalysis.

### 4.1.1 Guessing a good linear approximation

Since the nearly linear component of this instantiation of the cipher resides in the substitution function, we should focus on it. If we look carefully at the substitution table, we can see that its values somewhat resemble the output of the substitution function used for the linear instantiation. Indeed, using the notation of the project instructions, we can rewrite the substitution table as the following piecewise function.

$$\forall j \in \{1, \dots, \ell\},\ y_i(j) = \begin{cases} 10v_i(j) \mod p & \text{if } v_i(j) = 3 \mod p \\ 7v_i(j) \mod p & \text{if } v_i(j) = 4 \mod p \\ 2v_i(j) \mod p & \text{otherwise} \end{cases} \tag{4}$$

We notice that in 9 cases out of 11, this function is identical to the one used for the linear instantiation of the cipher. Therefore, we choose to use the substitution function of the linear instantiation to find a linear approximation of the nearly linear one. But now we have again the linear instantiation, so the corresponding matrices are the same as those in (2).

To evaluate this approximation, we follow these steps:

1. We generate $N$ random plaintexts and corresponding encryption keys, where $N$ is large (e.g., 1 million).
2. We encrypt each plaintext with its corresponding encryption key, using both the approximation matrices and the true encryption function. Whenever both result in the same ciphertext, we increment a counter $s$.
3. Finally, we compute the estimated success probability of the approximation as $s$ divided by $N$.

We may also compute the ratio of the estimated success probability with respect to correctly guessing a random ciphertext. If $\ell_x$ is the length of the ciphertext, this ratio is as follows.

$$\frac{s/N}{1/p^{\ell_x}}$$

### 4.1.2 Performing a full linear cryptanalysis

We can perform a linear cryptanalysis of the nearly linear instantiation of the cipher following a procedure similar to the one proposed by Matsui et al. We consider identical S-boxes for each state $p$-it (i.e., an element of the state vector).

First, we compute the bias for all possible relations of the following kind.

$$\alpha x + \beta\, \text{S-box}(x) = 0 \tag{5}$$

In our case, there are 10 relations with a bias of $\varepsilon = 8/11$ and one, the trivial one, with a bias of $\varepsilon = 10/11$.

Now, we go through the whole cipher until the second to last round and write down all the relations for the active S-boxes. In doing this, we carefully choose the appropriate relations (5) so that all the intermediate variables cancels out. To assist us with this daunting task, we draw the entire architecture of the cipher (see Figure 1, and note that the transposition operation and the linear transformation are merged together in the drawing).

Then, we combine the expressions of each round, and after doing all the computations we obtain the following expression, which, as expected, does not contain any intermediate variable, where $u_i$ is the $i$-th $p$-it of the plaintext and $v_5^i$ is the $i$-th input to the $i$-th S-box of the 5th round.

$$5u^1 + 5u^4 + 7u^5 + 7u^8 + 7v_5^1 + 7v_5^4 + 2v_5^5 + 2v_5^8$$

Finally, employing a Matsui-like algorithm, we should be able to obtain 4 out of 8 $p$-its of the 6th round key, which in turn would be 4 out of 8 $p$-its of the full key. Unfortunately, because of how the round keys are constructed (i.e., by appending a 4 $p$-its subkey to itself) we would actually obtain only 2 $p$-its instead of 4. This means that we would then need to brute-force the remaining 6 $p$-its, which require too much computing time for us (but definitely not much for a patience attacker!).

The complete computations of the linear cryptanalysis are available in attachment 1, "Linear cryptanalysis." Beware that, because of time constraints, that document does not contain full explanations in words but almost only the computations and a couple of algorithms.

## 4.2 Results

Concerning the first approach, for the reasons explained previously, the matrices $A$ and $B$ are the same as those in (2), and the matrix $C$ is the identity matrix. The corresponding probability (estimated with a local desktop computer on 1 million random plaintext encryptions) is

$$p_{\text{approx}} \stackrel{\text{def}}{=} \mathrm{P}[Ak + Bu + Cx = 0] \approx 0.0332\,\%,$$

which is much greater than the one of a random guess of the ciphertext,

$$p_{\text{randguess}} \stackrel{\text{def}}{=} \frac{1}{p^{\ell_x}} = \frac{1}{11^8} \approx 0.000000466\,\%.$$

Indeed, we have

$$\frac{p_{\text{approx}}}{p_{\text{randguess}}} \approx 7.12 \cdot 10^4 \implies p_{\text{approx}} \gg p_{\text{randguess}}.$$

Regarding the second approach, unfortunately we couldn't get any concrete results beyond those already mentioned in the explanation above.

# 5 Task 8

## 5.1 Description

A meet-in-the-middle attack is similar to a brute-force attack, and the crucial aspect is that we perform it on a reduced key space because of how the cipher realizes the encryption.

The plaintext-ciphertext pairs at our disposal are encrypted twice using the same non-linear instantiation of the cipher with two different keys. In practice, this does not increase significantly the key space size, which turns out to be $11^4 + 11^4$.

Given a plaintext-ciphertext pair, to perform the meet-in-the-middle attack, we follow these steps:

1. Generate two lists of all possible $11^4$ keys: a list of encryption keys and a list of decryption keys.
2. Encrypt the plaintext with all the encryption keys, obtaining $11^4$ ciphertexts.
3. Decrypt the ciphertext with all the decryption keys, simultaneously checking whether the resulting plaintext is equal to one of the $11^4$ ciphertexts, and if so, adding the corresponding encryption and decryption keys to a list of candidate keys.
4. Finally, going through all candidate keys, check which of those both correctly encrypt the original plaintext to the original ciphertext and vice versa. That will be the correct key.

Since we have at our disposal five plaintext-ciphertext pairs, and we know that they are all encrypted with the same key, we perform the meet-in-the-middle attack on the first pair. Then, we check the correctness of each candidate key (point 4 of the above procedure) against all five pairs to be sure we are not making any mistakes.

## 5.2 Results

For each of the five plaintext-ciphertext pairs, the pair of keys recovered using a meet-in-the-middle attack on the non-linear instantiation of the cipher is the following.

$$\hat{k}' = [10, 6, 1, 6] \quad \hat{k}'' = [2, 8, 9, 3]$$

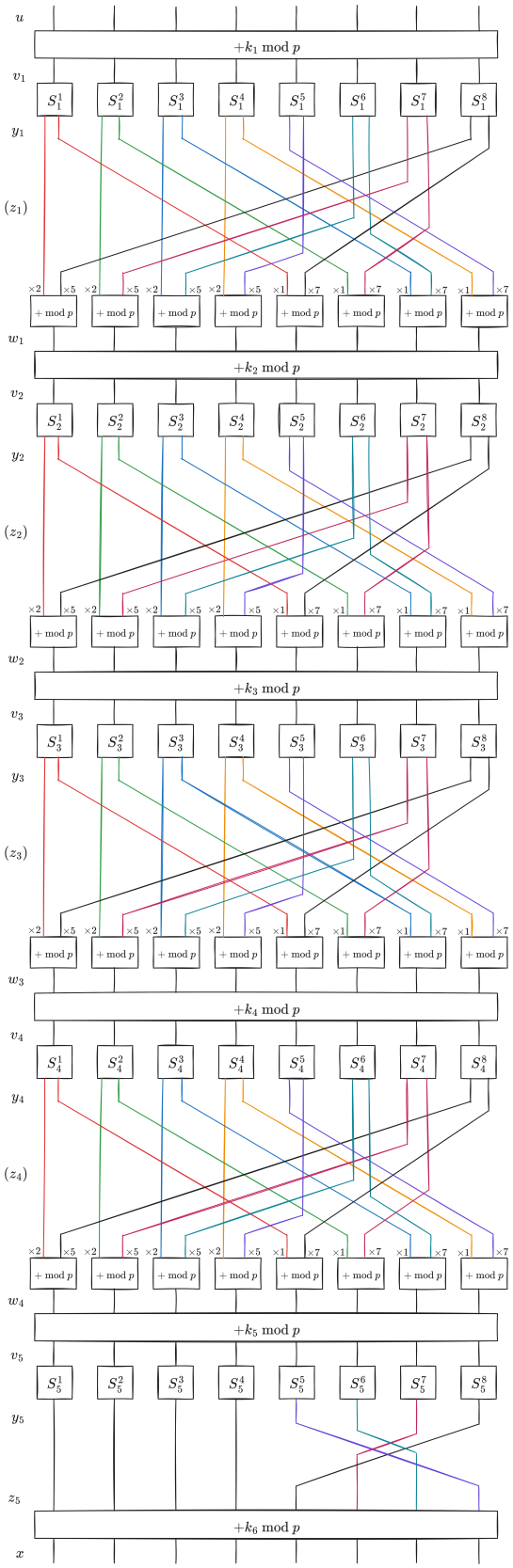


Figure 1: Simplified AES-like cipher complete architecture.