Re-implementation of the Picnic signaturescheme in Python

Thorsten Knoll

info@thorstenknoll.de

March, 2019

I. Introduction

Post-quantum cryptography, NIST and Picnic

Quantum computers are experiencing fast development and seem to be available within a timeframe of the next few decades. One of their properties will be to break huge parts of modern cryptography. Especially the discrete logarithm and prime-factorisation loose their trapdoor funcionality in regards to the efficient quantum algorithms from Grover and Shor. Therefore the need for new cryptographic algorithms arises, beeing save in regards to the availability of quantum computers. This field of research goes with the name "Post-quantum cryptography" (PQC). The american National Institute of Standards and Technology (NIST) called out a challenge to find the next PQC standards. This challenge is in round two of three at the time of writing this document. 69 submissions from round one were reduced to 26 candidates in round two of the challenge. These 26 candidates got announced by NIST not long ago at the end of january 2019. One of the submissions surviving the first round is the Picnic signaturescheme.

Evaluation and categorization

Learning PQC

Reference documents and implementations by the Picnic Team

II. LowMC

Idea and overview

LowMC is a blockcipher in a roundbased construction scheme like many other blockciphers. The name is an abbreviation for "Low multiplicative complexity". XOR in GF2 is a linear operation (ADD), while the multiplication (AND) in GF2 is a non-linear operation (Figure 1). LowMC tries to keep the count of AND operations as low as possible while still maintaining a given security level (L1, L3, L5). In Picnic, the low mutliplicative complexity of LowMC is used for reducing the signature lengths.

A	B	XOR(A,B)	AND(A,B)
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

Figure 1: Linear and non-linear operations

Figure 2 shows the roundbased scheme of LowMC. Only the sbox part of the algorithm uses multiplications (ANDs). The other parts strictly contain only linear operations.

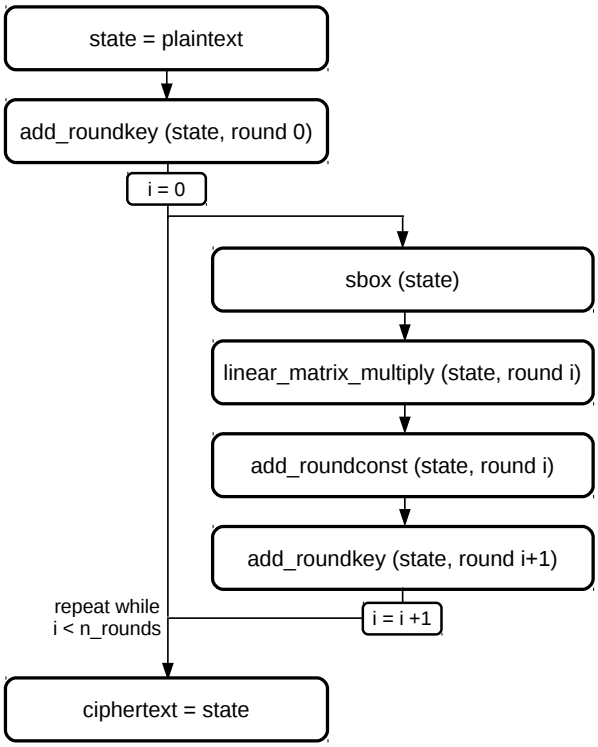

Figure 2: LowMC scheme

Pre-calculated constants

The python program `generator.py` generates the files for all security levels with the following pre-calculated constants:

- Linear layer matrices
- Round constants
- Roundkey matrices

The generation of the constants-files is not mandatory for usage as the project contains them "ready to use".

Private functions

`__apply_sbox()`

`__multiply_with_lin_mat(round)`

`Add roundconstant`

`__multiply_with_lin_mat(round)`

`Decryption functions`

Public functions (API)

`LowMC(Security level) - Constructor`

Constrcuts an object of LowMC with the parameters regarding to the security level. The following security levels are available and shall be given to the constructor as strings: "picnic-L1", "picnic-L3" and "picnic-L5". The fitting file with the constants must be in the project directory and gets read (see Pre-calculated constants).

`generate_priv_key()`

Generates a private key of the length specified within the security level. The private key is stored in a private variable. The python package CSPRNG `os.urandom(bytelength)` is used.

`set_priv_key(priv_key)`

Instead of generating the private key it can also be set by giving a bytearry to this function. The bytearray must match the specified keylength from the security level.

`encrypt(plaintext)`

Encrypts a plaintext and returns the ciphertext. The plaintext length must match the specified blocksize length (security level) and must be given as a bytearray to the function. The ciphertext is returned as a bytearray of the same length. Before using this function a private key must be set (or generated).

`decrypt(ciphertext)`

Decrypts a ciphtertext and returns the plaintext. The ciphertext length must match the specified blocksize length (security level) and must be given as a bytearray to the function. The plaintext is returned as a bytearray of the same length. Before using this function a private key must be set (or generated).

Testvectors

III. Picnic

References

[1] Microsoft *Picnic PQC Signatureschene*. 2018