Pipeline for generating HAWK key pairs We want to analyze and do experiments on the HAWK digital signature scheme. To this end, we use a version of HAWK implemented in Python, written by the original authors of HAWK. Their original code can be found at https://github.com/hawk-sign/hawk-py . To suit our needs, some changes and modifications will be done to the implementation, permitted by their license. (see license.txt in their github).

Their implementation supports key generation of purposed bit-security levels 64, 128 and 256, respectively titled HAWK-256, HAWK-512, and HAWK-1024. The latter two security levels aims to target NIST-I and NIST-V security levels, while a bit-security of 64 is too low for real-life use. Regardless, as stated in the HAWK specification paper, breaking the lowest security level is not trivial, and could potentially lead to some interesting consequenses for the future of HAWK.

First, we implement a pipeline for generating (several) public/private key pairs. To be able to reproduce results, we also incorporate a seeding function. This enables us to deterministically create the same collection of key pairs.

A quick note on the Python implementation: This implementation by the HAWK team was made to support the specifications paper, and is not suited for production, and is not as efficient as their optimized C implementation (https://github.com/hawk-sign/dev). However, since we are interested in doing experiments, we use Python for its simplicity, readablility, and rich selection of libraries. In later stages of this thesis, we might want to run large(r) scale experiments which relies on optimized implementations. In this case, we will either implement the experiments in a faster, compiled language (e.g. C/C++), or, if possible, use Python libraries and packages which are fast and optimized.

NTRU Solve For the analysis of the HAWK digital signature scheme, we are particularly interested in the algorithm *NTRU Solve*, which given polynomials f, g, returns F, G such that the equation $fG - gF = q \mod \phi$ holds. In HAWK, $q = 1$ and $\phi = x^n + 1$.

The matrix $\mathbf{B} = \begin{pmatrix} f & F \\ g & G \end{pmatrix}$ is the lattice secret basis used for creating signatures.

Generally, the NTRU equation is not trivial to solve. However, in [2] a suitable algorithm is presented to this end.

The algorithm [1]

References

[1] Hawk version 1.0.1. 2023. Accessed: [12.04.2024].

[2] Thomas Pornin and Thomas Prest. More efficient algorithms for the ntru key generation using the field norm. *PKC 2019, part II*, 11443, 2019.