Category: Cryptanalysis
Task: Wrong Ring
Description: Is post-quantum cryptography too **complex** for you?
Author: Miruna
Points: 936/1000

Idea: This is an attack on (non-dual) Ring Learning With Errors (RLWE) which works by interpreting the RLWE samples as Polynomial Learning With Errors (PLWE) samples.

Why it works: the corresponding PLWE error is small enough in the last components and you can recover the secret using linear algebra.

The flag is hidden into an image which has been encrypted with AES CFB with a secret key. Finding the key reduces to finding a secret polynomial $s(x)$ of degree $n = 256$ with integer coefficients modulo $p = 1487$.

In the file `challenge.txt`, we are given the evaluations at the roots of the polynomial $f(x) = x^{256} + 1486$ of 8 pairs of polynomials $(a_i(x), b_i(x))$, where the $a_i(x)$'s were sampled uniformly random in $\frac{\mathbb{Z}_p[x]}{(f)}$ and $b_i(x) = a_i(x) \cdot s(x) + e_i(x)$ modulo $(p, f(x))$ for some secret 255 degree error polynomials $e_i(x)$ with real coefficients.

We can actually recover the polynomials $a_i(x)$ and $b_i(x)$ by interpolation in the roots of the polynomial f. Note that the multiplication by the Vandermonde matrix `Sigma` defined in the script corresponds to the evaluation of a polynomial at the roots of f, which means that the multiplication by its inverse `Sigma_inv` corresponds exactly to the interpolation operation.

Once we have recovered the polynomials $a(x)$ and $b(x)$, we may write the following system of equations:

$$Rot_f(a) \cdot (s_0, \ldots, s_{n-1})^t = (b_0, \ldots, b_{n-1})^t - (e_0, \ldots, e_{n-1})^t,$$

where $Rot_f(a)$ is the $n \times n$ matrix which corresponds to the multiplication by the polynomial $a(x)$ modulo $f(x)$ with respect to the polynomial basis $1, x, \ldots, x^{n-1}$ and s_i (resp. b_i, e_i) are the coefficients of $s(x)$ (resp. $b(x), e(x)$) with respect to the same basis. The $Rot_f(a)$ matrix is invertible with high probability over the uniform choice of $a(x)$. If the error vector didn't exist, we could recover the secret $s(x)$ by Gaussian elimination using only one sample $(a(x), b(x))$. Now let's look deeper in the way the error was generated.

At the beginning, we are given RLWE samples, which implies that for each pair $(a(x), b(x))$, the error polynomial $e(x)$ is generated in the *canonical embedding*: first, each component of a 256-dimensional vector is sampled independently from a Gaussian distribution with standard deviation $\sigma = 1000$. Then, the vector is multiplied by the matrix `U`. What we get is the vector of values of $e(x)$ in the roots of the polynomial $f(x)$. When we multiply by `Sigma_inv`, we look at the error in the *coefficient embedding* as in the PLWE setup. What happens here is pretty nice: the error we have just described gets distorted and the distorsion is measured by the singular values of the matrix `Sigma_inv` $\cdot$ `U`. In [1] an explicit formula for the singular values of `Sigma_inv` $\cdot$ `U` has been given. The singular values decrease geometrically and for the values given in the task, the coefficient of x^i of the polynomial $e(x)$ is less than 1/2 in absolute value for $223 < i <= 255$.

We now go back to our system $Rot_f(a) \cdot (s_0, \ldots, s_{n-1})^t = (b_0, \ldots, b_{n-1})^t - (e_0, \ldots, e_{n-1})^t$. We round the coefficients in the right-hand side and we actually remove the errors of high index. By collecting 32 exact equations for each pair $(a(x), b(x))$, we can build a linear system of 256 equations with 256 unknowns and solve it in order to find $s(x)$.

Take away: We should pay attention to the way we sample the error for post-quantum cryptographic schemes whose security is based on RLWE or PLWE. Bad sampling may lead to insecure instantiations.

Reference: [1] `https://www.esat.kuleuven.be/cosic/publications/article-2637.pdf`