Message Recovery Attack on ACES

Samuel Jaques

January 26, 2024

`https://arxiv.org/abs/2401.13255` (hereinafter "the paper") is an interesting and ambitious attempt to use category theory to unify homomorphic properties in different cryptographic schemes. It also proposes a bootstrapping-free FHE scheme called ACES, which unfortunately has an efficient message recovery attack, detailed below.

1 ACES

Translating the ACES cryptosystem into more familiar terms for lattice cryptographers, it is similar to a module LWE scheme. The usual notation for module LWE is with a ring

$$R_q = \mathbb{Z}_q[x]/(u(x)). \tag{1}$$

The paper uses $\mathbb{Z}_q[x]_u$ to refer to this ring. The scheme is parameterized by q, u, a root ω such that $u(\omega) \equiv 0 \mod q$, and module parameters k_1 and k_2. This is in fact a slight generalization of the scheme, which used $k_2 = 1$.

KeyGen: Alice selects a random matrix $A \in R_q^{k_2 \times k_1}$, a secret vector $s \in \mathbb{R}_q^{k_1}$, and a secret error $e \in R_q^{k_2}$, but chosen specifically so that each component e_i satisfies $e_i(\omega) \equiv 0 \mod q$. She outputs an LWE public key $(A, b = As + e)$ and retains s as the secret key.

(in the notation of the paper, $f_0 = A$, $f_1 = b$, $x = s$, $e = e$).

Encrypt: Bob selects a random vector of polynomials $r \in R_q^{k_2}$, a random error $e' \in R_q$ such that $e'(\omega) \equiv 0 \mod q$, and a random message encoding polynomial I'll call $n \in R_q$, chosen such that $n(\omega) \equiv m \mod q$, where m is the message he wants to encrypt. He computes $c_1 = r^T A \in R_q^{k^2}$ and $c_2 = r^T b + e' + n \in R_q$ and outputs (c_1, c_2).

(in the notation of the paper, $b = r$ and $r(m) = n$; e' is not apparent in the paper but is in the python implementation).

Decrypt: Alice receives c_1 and c_2 and computes $c_2 - c_1 s \in R_q$ and evalutes this polynomial at ω. The result will be $m \mod q$.

1.1 Break

In the implemented scheme with $k_2 = 1$, r is not hidden by the matrix A, which is wider than it is tall. We can recover r by taking the first invertible element of A, inverting it, and multiplying it by the corresponding element of c_1.

For larger parameterizations (e.g., if $A \in R_q^{k_2 \times k_1}$ for $k_2 \leq k_1$), it is always straightforward to solve for r since $c_1 = r^T A$ is a linear system of equations.

We might be tempted to fix the problem by setting $k_2 > k_1$, so that A is "tall" and r will be hidden as a module-SIS problem. Unfortunately, this permits an efficient key recover attack.

An attacker can evaluate all public terms at the point ω. Since evaluation at ω is a homomorphism modulo $u(x)$ (as ω is a root), then the attacker transforms the public key to $(A(\omega), A(\omega)s(\omega))$. Notice that $A(\omega) \in \mathbb{Z}_q^{k_2 \times k_1}$ and $A(\omega)s(\omega)$ is a vector in $\mathbb{Z}_q^{k_2}$: again, this is a (overdetermined) linear system and we can solve for $s(\omega)$.

Obtaining $s(\omega)$ does not give the attacker s, but they do not need s. Since decryption is $(c_2 - c_1 s)(\omega)$, and evaluating at ω is a homomorphism, one can compute $c_2(\omega) - c_1(\omega)s(\omega)$, and this will reveal m.

The goal of the scheme seems to be to avoid errors in LWE decryption by choosing polynomials which evaluate to 0 on a known point. However, evaluating at that point will always reduce the problem to traditional matrix-vector LWE over $\mathbb{Z}_q$, but in such a way that the public key and the ciphertext do not have errors. Thus, at most one side (the ciphertext c_1 or the public key b) will be underdetermined, and we can recover the other side.