“Encrypt-then-MAC” with Kyber/ML-KEM is insecure

Ganyu Xu

July 12, 2024

After some additional thoughts I found a chosen-ciphertext attack against the “Encrypt-then-MAC” transformation when combined with Kyber/ML-KEM. This attack takes advantage of the general structure of LWE-based cryptosystem, and I have no immediate ways to patch this problem. Unfortunately I think this means that `EtM` is a dead end with lattice-based schemes.

1 A plaintext-checking attack against Kyber

Recall the construction of `Kyber.CPAPKE`

Algorithm 1 $\texttt{KeyGen}_{\texttt{PKE}}$

1: $A \xleftarrow{\$} R_q^{k \times k}, \mathbf{s} \xleftarrow{\$} \mathcal{X}_{\eta_1}^k$
2: $\mathbf{t} \leftarrow A \cdot \mathbf{s}$
3: $\texttt{pk} \leftarrow (A, \mathbf{t}), \texttt{sk} \leftarrow \mathbf{s}$
4: **return** $(\texttt{pk}, \texttt{sk})$

Algorithm 2 $\texttt{E}_{\texttt{PKE}}(\texttt{pk}, m)$

1: $(A, \mathbf{t}) \leftarrow \texttt{pk}$
2: $\mathbf{r}_1 \xleftarrow{\$} \mathcal{X}_{\eta_1}^k$
3: $\mathbf{e}_1 \xleftarrow{\$} \mathcal{X}_{\eta_2}^k, e_2 \xleftarrow{\$} \mathcal{X}_{\eta_2}$
4: $\mathbf{c}_1 \leftarrow A^\intercal \cdot \mathbf{r}_1 + \mathbf{e}_1$
5: $c_2 \leftarrow \mathbf{t}^\intercal \cdot \mathbf{r}_1 + e_2 + m \cdot \lceil \frac{q}{2} \rfloor$
6: **return** $(\mathbf{c}_1, c_2)$

Algorithm 3 $\texttt{D}_{\texttt{PKE}}(\texttt{sk}, c)$

1: $(\mathbf{c}_1, c_2) \leftarrow c$
2: $\mathbf{s} \leftarrow \texttt{sk}$
3: $\hat{m} = c_2 - \mathbf{s}^\intercal \cdot \mathbf{c}_1$
4: $\hat{m} \leftarrow \text{Round}(\hat{m})$
5: **return** $\hat{m}$

Figure 1: `PKE` routines

This construction has no ciphertext integrity, meaning that an adversary can submit well-formed AND malformed ciphertexts and recover the secret key by observing the behavior of the decryption routine. Here we present a plaintext-checking attack, which uses a plaintext checking oracle:

Algorithm 4 $\texttt{PCO}(m, c)$

1: **return** $[\![\texttt{D}(\texttt{sk}, c) = m]\!]$

Also recall that with Kyber/ML-KEM, each polynomial can be transformed into the NTT domain:

$$R_q = \frac{\mathbb{Z}_{3329}[x]}{\langle x^{256} + 1 \rangle} \cong \frac{\mathbb{Z}_{3329}[x]}{\langle x^2 + \zeta \rangle} \times \frac{\mathbb{Z}_{3329}[x]}{\langle x^2 + \zeta^3 \rangle} \times \ldots \times \frac{\mathbb{Z}_{3329}[x]}{\langle x^2 + \zeta^{255} \rangle}$$

Where ζ is any solution to $\zeta^{128} + 1 \equiv 0 \mod 3329$ (Kyber picked $\zeta = 17$). We denote the NTT representation by:

$$\text{NTT}(y \in R_q) = (\hat{y}_1, \hat{y}_2, \ldots, \hat{y}_{128})$$

Where each $\hat{y}_j$ for $1 \leq j \leq 128$ is a degree-1 polynomial.

For each of $i \in \{1, 2, \ldots, k\}$ and each of $j \in \{1, 2, \ldots, 128\}$, an adversary can craft a maliciously malformed ciphertext $c = (\mathbf{c}_1, c_2)$ such that:

- $\mathrm{NTT}(c_2) = (0, 0, \ldots, \hat{c}_{2,j}, \ldots, 0)$ is all 0's except for the j-th entry, which is chosen by the adversary
- $\mathbf{c}_1 = (0, 0, \ldots, c_{1,i}, \ldots, 0)$ is all 0's except for the i-th entry $c_{1,i}$, whose NTT representation $\mathrm{NTT}(c_{1,i}) = (0, 0, \ldots, \hat{c}_{1,i,j} = 1, \ldots, 0)$ is all 0's except for the j-th entry, which is 1.

In line 3 of the decryption routine (algorithm 3):

$$\begin{aligned}
\mathrm{NTT}(c_2 - \mathbf{s}^\intercal \cdot \mathbf{c}_1) &= \mathrm{NTT}(c_2) - \mathrm{NTT}(\mathbf{s}^\intercal \cdot \mathbf{c}_1) \\
&= (0, 0, \ldots, \hat{c}_{2,j}, \ldots, 0) - \mathrm{NTT}(s_i \cdot c_{1,i}) \\
&= (0, 0, \ldots, \hat{c}_{2,j}, \ldots, 0) - \mathrm{NTT}(s_i) \circ \mathrm{NTT}(c_{1,i}) \\
&= (0, 0, \ldots, \hat{c}_{2,j}, \ldots, 0) - (\hat{s}_{i,1}, \hat{s}_{i,2}, \ldots, \hat{s}_{i,128}) \circ (0, 0, \ldots, c_{1,i,j} = 1, \ldots, 0) \\
&= (0, 0, \ldots, \hat{c}_{2,j} - \hat{s}_{i,j}, \ldots, 0)
\end{aligned}$$

I will make an unverified but probably correct claim: *if $\hat{c}_{2,j} - \hat{s}_{i,j} \neq 0$ then with very high probability $\mathrm{NTT}^{-1}((0, 0, \ldots, \hat{c}_{2,j} - \hat{s}_{i,j}, \ldots, 0))$ will not round to* 0. This means that with very high probability, $(\mathbf{c}_1, c_2)$ will not decrypt to 0 if $\hat{c}_{2,j} \neq \hat{s}_{i,j}$, which is equivalent to $\mathtt{PCO}(m = 0, c = (\mathbf{c}_1, c_2)) = 0$.

The adversary can thus iterate through all q^2 possible degree-1 polynomials to find the correct value for $\hat{s}_{i,j}$, then repeat it for all i, j. In $q^2 \cdot k \cdot \frac{n}{2}$ operations, the adversary can recover the secret key.

2 EtM is vulnerable to the key recovery attack above

Suppose an adversary crafts a malicious ciphertext using the strategy described above $c = (\mathbf{c}_1, c_2)$, if the secret key value is such that $\hat{c}_{2,j} = \hat{s}_{i,j}$, then $m = 0$ should be the correct decryption, which means that $k = G(0)$ should be the correct MAC key, so the adversary computes the tag $t = \mathtt{MAC}(G(0), c)$.

If (c, t) is rejected, then the adversary learns that 0 is not the correct decryption. From here the key recovery attack described above can be executed. A similar attack can be executed using `PCO` against `EtM`. This means that at least with Kyber/ML-KEM, `EtM` is not one-way secure with either `PCO` or `CVO`.