Question 6

Recall the parameters of Kyber:

- $R_q = \mathbb{Z}_q[x]/p(x)$ for $q = 3329$ and $p(x) = x^{256} + 1$.
- $A \in R_q^{k \times k}$ for $k = 2, 3, 4$ depending on the security level. $\mathbf{s}, \mathbf{e}, \mathbf{b} \in R_q^k$
- $m \in R_q$ but each coefficient is either 0 or 1

Also recall that for $p(x) = x^{256} + 1$, the NTT representation of each $f(x) \in R_q$ consists of 128 degree-1 polynomials in $\mathbb{Z}_q[x]$:

$$\text{NTT}(f) = (\hat{f}_0 + \hat{f}_1 x, \hat{f}_2 + \hat{f}_3 x, \ldots, \hat{f}_{254} + \hat{f}_{255} x)$$

Finally, we claim without proof that:

1. Polynomial addition and multiplication in R_q are all performed coordinate-wise
2. $\text{NTT}(f) = \mathbf{0}$ if and only if $f = 0$

For some $\mathbf{s} \in R_q^k$, let $\mathbf{s}[i] = R_q$ denote the i-th polynomial in the vector, and $\mathbf{s}[i][j] = s_0 + s_1 x$ denote the j-th entry of the NTT representation of $\mathbf{s}[i]$

For some $l \in \{1, 2, \ldots, k\}$ and some $j \in \{1, 2, \ldots, 128\}$ we can recover the j-th entry in the NTT representation of the l-th polynomial in $\mathbf{s} \in R_q^k$ in q^2 steps.

1. Set $c_2 \in R_q$ to have NTT representation $\text{NTT}(c_2) = (0, 0, \ldots, q(x), \ldots, 0)$ where all entries are 0 except for the j-th entry, which is some random guess $g(x) = g_0 + g_1 x \in \mathbb{Z}_q$.
2. Set $\mathbf{c}_1 \in R_q^k$ such that all except for the l-th polynomial is 0, and all but the j-th entry to the NTT representation of the l-th polynomial is exactly 1
3. Feed $c = (\mathbf{c}_1, c_2, H(0))$ (aka I expect $m = 0$) to the decapsulation oracle and get $K = \mathcal{O}^D(c)$
4. If $K = H(0, (\mathbf{c}_1, c_2), 0)$, then $g(x)$ is the correct guess for the j-th entry of the NTT representation of the l-th polynomial in $\mathbf{s} \in R_q^k$. This works because $\text{NTT}(\mathbf{c}_1)$ is all 0 except for the j-th entry in the l-th polynomial in NTT domain, which means $\text{NTT}(\mathbf{c}_1 \cdot \mathbf{s}) = \text{NTT}(\mathbf{c}_1) \circ \text{NTT}(\mathbf{s}) = (0, 0, \ldots, \mathbf{s}[l][j], \ldots, 0)$, so $\text{NTT}(c_2 - \mathbf{c}_1 \cdot \mathbf{s}) = (0, 0, \ldots, g(x) - \mathbf{s}[l][j], \ldots, 0)$. I claim without proof that if $g(x) - \mathbf{s}[l][j] \neq 0$, then $c_2 - \mathbf{c}_1 \cdot \mathbf{s}$ will not round to $0 \in R_q$, which means that the decryption $\hat{m} \neq 0$, which will cause the decapsulation oracle to output the implicit rejection $H(\mathbf{s}, c, 1)$ instead of $H(0, (\mathbf{c}_1, c_2), 0)$

Since $g(x)$ is a degree-1 polynomial, it takes at most q^2 guesses to recover a single entry. We can then repeat the attack for all $j \in \{1, 2, \ldots, 128\}$ and $l \in \{1, 2, \ldots, k\}$. In other words, we can recover the secret key $\mathbf{s}$ (by recovering its NTT representation) in $q^2 \cdot \frac{n}{2} \cdot k$ steps.

Comparing to McEliece

1. Classic McEliece checks that the norm (Hamming norm in this case) of the error term is exactly t as defined by the security parameters. Any codeword with strictly fewer than t bits of errors can still be correctly decoded, but the decryption routine will detect that the ciphertext has been tempered with and will appropriately reject the ciphertext as invalid.
2. Kyber will accept any ciphertexts whose error term's norm is no more than the threshold for decryption failure, which allows substantially more ciphertext malleability. This increased tolerance for ciphertext malleability is what makes the IND-CCA attack above work for Kyber, but not for McEliece.