Question 2

(1)

Here is the decryption algorithm

Algorithm 1: Compressed McEliece decryption

Input: $c \in \mathbb{F}_2^{n-l}$, $\text{sk} = (P, \mathcal{C}, S)$

1 Pad the low-order bits of c with 0's such that $c' = [c, 0, 0, \ldots, 0] \in \mathbb{F}^n$;

2 $c'' \leftarrow P^{-1}c'$;

3 $\hat{m} \leftarrow \mathcal{C}.\text{decode}(c'')$;

4 $\hat{m} \leftarrow S^{-1}\hat{m}$;

5 **return** $\hat{m}$;

Among the high-order bits of c'', there are exactly $t - l$ bits of error, introduced by the sampled error term e. Among the low-order l bits of c'', there are up to l bits of errors since the padded 0's are blind guesses. Therefore, there are up to t bits of errors in c'', and since $t \leq \frac{d-1}{2}$, the (n, k, d)-code is guaranteed to correct the error and recover the true m.

(2)

The vanilla McEliece encryption scheme outputs the entire (partially corrupted codeword) c, which takes n bits. This compressed scheme discarded the low-order l bits, so the ciphertext takes $n - l$ bits, which saves space by a factor of $\frac{l}{n}$

(3)

We already know that the McEliece encryption scheme is not IND-CPA. Instead, we will estimate the difficulty of breaking the encryption scheme by computing the probability of correctly decrypting some ciphertext without using the secret key. Specifically, since $c = Am + e$ where $A \in \mathbb{F}^{n \times k}$ is an overdetermined linear system, if the adversary can recover e, then it can recover m.

In the un-compressed McEliece scheme, the encryption routine corrupts exactly t out of n bits. There are a total of $\binom{n}{t}$ possible error terms to choose from. Without knowing additional information, the adversary can do no better than a blind guess:

$$\epsilon_0 = \frac{1}{\binom{n}{t}}$$

In the compressed scheme, the encryption routine corrupts $t - l$ out of the high-order $n - l$ bits. The low-order l bits are blind guesses. There are a total of $\binom{n-l}{t-l} \cdot 2^l$ possible error values to choose from:

$$\epsilon_1 = \frac{1}{\binom{n-l}{t-l} \cdot 2^l}$$

Observe that

$$\begin{aligned}
\frac{\epsilon_0}{\epsilon_1} &= \frac{\binom{n-l}{t-l} \cdot 2^l}{\binom{n}{t}} \\
&= \frac{t \cdot (t-1) \cdot \ldots \cdot (t-l+1)}{n \cdot (n-1) \cdot \ldots \cdot (n-l+1)} \cdot 2^l
\end{aligned}$$

Since $t = \frac{d-1}{2} \leq \frac{d}{2} < \frac{n}{2}$, we know $\frac{t}{n} \leq \frac{1}{2}$. Furthermore, we claim without proof that if $0 < a < b$ then $\frac{a}{b} \geq \frac{a-1}{b-1}$, which means

$$\frac{1}{2} > \frac{t}{n} > \frac{t-1}{n-1} > \ldots > \frac{t-l+1}{n-l+1}$$

Therefore:

$$\frac{\epsilon_0}{\epsilon_1} = \frac{t \cdot (t-1) \cdot \ldots \cdot (t-l+1)}{n \cdot (n-1) \cdot \ldots \cdot (n-l+1)} \cdot 2^l \leq 1$$

This means that a blindly-guessing adversary against the compressed scheme has higher advantage than a blindly-guessing adversary against the vanilla scheme. In other words, the compressed scheme is less secure.

(4)

For a linear amount of reduction in ciphertext size, we lose an exponential amount of security. This tradeoff is not worth it.