Appendix A

Proof of Theorem 1

Proof: Given an order preserving function $y_i = f(x_i) + r_i$, $\forall x_i, x_j$, if we have $y_i + y_j \in [f(x_i + x_j), f(x_i + x_j) + r_{i+j}]$, obviously, $y_i$ is also additive order preserving. Therefore, our goal is reduced to prove that $\forall x_i, x_j, y_i + y_j \in [f(x_i + x_j), f(x_i + x_j) + r_{i+j}]$. Without loss of generality, we assume $x_i \leq x_j$. Then we have $f(x_i + x_j) = f(x_i) + \Delta f(x_j) + \cdots + \Delta f(x_j + x_j - 1)$, and hence $f(x_i) + f(x_j) = 2f(x_i) + \Delta f(x_i) + \cdots + \Delta f(x_j - 1)$. Therefore, we have $y_i + y_j - f(x_i + x_j) \geq f(x_i) + f(x_j) - f(x_i + x_j)$ $\geq f(x_i) - \Delta f(x_i) - \cdots - \Delta f(x_i + x_j - 1)$ $\geq r_{i+j} = r_{i+j}$ $\geq 0$ (17)

Additionally, we have $y_i + y_j - f(x_i + x_j) - r_{i+j} \geq f(x_i) + f(x_j) - f(x_i + x_j) + r_{i+j}$ $\geq f(x_i) - \Delta f(x_i) - \cdots - \Delta f(x_i + x_j - 1)$ $\geq f(x_i) - \Delta f(x_i) - \cdots - \Delta f(x_i + x_j - 1)$ $\geq r_{i+j} = r_{i+j}$ $\geq 0$ (18)

From the above two equations, we can easily get $\forall x_i, x_j, y_i + y_j \in [f(x_i + x_j), f(x_i + x_j) + r_{i+j}]$. Up to now, Theorem 1 is proved.

Appendix B

Proof of Theorem 2

Given the DBDH (Decisional Bilinear Diffie-Hellman) assumption, PRMSM is semantically secure against the chosen keyword attack.

Proof: Assume a polynomial-time adversary $A$ has a non-negligible advantage $\epsilon$ against PRMSM. Then we can build a simulator $B$ that solves DBDH with advantage $\epsilon/2$. The challenger flips a fair coin $\delta$ outside of $B$'s view. If $\delta = 0$, he sends $(A, B, C, Z) = (g^\alpha, g^b, g^c, g^{\alpha b})$ to $B$; otherwise he sends $(A, B, C, Z) = (g^a, g^b, g^c)$ to $B$, where $a, b, c, z \in Z_p$ are randomly generated. The goal of $B$ is to guess $\delta'$ for $\delta$ by interacting with $A$ and playing the following game.

Setup: $B$ generates his private key $(k_1, k_2)$, and sends the public key $(g, g^{k_1}, g^{k_2}, g^a, g^b, z)$ to $A$.

Phase 1: $B$ maintains a keyword list $L_w$, which is initially empty. $A$ can issue any keyword $w \in W$ and ask $B$ to generate the corresponding keyword ciphertext $\hat{w}$ for polynomial times. If $w \notin L_w$, $B$ adds $w$ to $L_w$ and sends $\hat{w}$ to $A$.

Challenge: $A$ sends two keywords $w_0$ and $w_1$ with equal length, where $w_0, w_1 \notin L_w$, to $B$, $B$ randomly sets $\mu \in \{0, 1\}$, computes the ciphertext $\hat{w}_\mu = (Z^H(w_\mu, k_2), g^{k_1, k_2}, Z)$, and sends $\hat{w}_\mu$ to $A$.

Phase 2: $A$ continues to submit keywords to request $B$ for generating the ciphertext of keyword as in Phase 1. The restriction here is that $w_0$ and $w_1$ cannot be submitted.

Guess: $A$ outputs its guess $\mu' \in \{0, 1\}$ for $\mu$. If $\mu' = \mu$, $\hat{w}_\mu$ is a correct encryption of $w_\mu$, then $B$ outputs $\delta' = 0$; otherwise, $B$ outputs $\delta' = 1$.

To complete the proof of Theorem 2, we now compute $B$'s advantage in solving DBDH. If $\delta = 0$, then $\hat{w}_\mu$ is a valid encryption of $w_\mu$, so $A$ will output $\mu' = \mu$ with probability $1/2 + \epsilon$. Additionally, if $\delta = 1$, i.e., $Z$ is randomly chosen, $A$ will output $\mu' = \mu$ with probability $1/2$. Therefore, $B$ will guess $\delta' = \delta$ with probability $1/2(1/2 + \epsilon + 1/2) = 1/2 + \epsilon/2$. That is, if the adversary $A$ has advantage $\epsilon$ against PRMSM, then the challenger $B$ will solve DBDH with advantage $\epsilon/2$.

Appendix C

Proof of Theorem 3

Given the DL assumption, PRMSM achieves keyword secrecy in the random oracle model.

Proof: We construct a challenger $B$ that plays the keyword secrecy game as follows.

Setup: $B$ generates the private key $k, k_1, k_2 \in Z_p$, and sends the public key $g, g^k, g^{k_1}, g^{k_2}$ to $A$.

Phase 1: $A$ adaptively queries the following oracle for polynomial times.

Oracle $O_1$: the challenger $B$ maintains an $O_1$-list, which is initially empty. Each entry of $O_1$-list is $<w, T_w>$. $A$ can query $O_1$-list for a keyword $w$, if $w$ is already in $O_1$-list, then $B$ returns $T_w$ to $A$, otherwise, $B$ generates the trapdoor $T_w$ for $w$, adds $<w, T_w>$ to $O_1$-list, and returns $T_w$ to $A$.

Challenge: $B$ chooses a keyword $w^*$ from the keyword dictionary uniformly at random, and returns the encrypted keyword $\hat{w}^* = (g^{k \cdot H(w^*), k_1, k_2}, g^{k_1, k_2}, g^{k \cdot r})$, and trapdoor $T_{w^*} = (g^{H(w^*)}, g^r)$ to $A$.
**Guess:** $A$ outputs its guess $w'$ for $w^*$, and sends $w'$ to challenger $B$. $B$ returns the encrypted keyword $\hat{w}'$ to $A$. If $\hat{w}'$ matches $T_{w^*}$, then $A$ wins the game.

To complete the proof of Theorem 3, we now compute $A$’s probability in winning the keyword secrecy game. Assume $A$ has already tried $t$ distinct keywords before outputting $w'$, then the size of remaining keyword dictionary is $u - t$. Additionally, due to the hardness of discrete logarithm, deriving $w^*$ from $\hat{w}'$ or $T_{w^*}$ is at most a negligible probability $\epsilon$, therefore, the probability that $A$ wins the keyword secrecy game is $\frac{1}{u-t} + \epsilon$. $\Box$