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 $x_i \le x_j$. Then we have $f(x_i+x_j)=f(x_i)+\Delta f(x_i)+\cdots+\Delta f(x_i+x_j-1)$, and $f(x_i) + f(x_j) = 2f(x_i)+\Delta f(x_i)+\cdots+\Delta f(x_j - 1)$. Therefore, we have

$$\begin{aligned} & y_i + y_j - f(x_i + x_j) \\ \ge\ & f(x_i) + f(x_j) - f(x_i + x_j) \\ =\ & f(x_i) - (\Delta f(x_j) + \cdots + \Delta f(x_i + x_j - 1)) \\ \ge\ & f(x_i) - i \cdot \Delta f(x_i) \\ =\ & r_{i\max} \\ >\ & 0 \end{aligned} \tag{17}$$

Additionally, we have

$$\begin{aligned} & y_i + y_j - f(x_i + x_j) - r_{(i+j)\max} \\ \le\ & f(x_i)+f(x_j)-f(x_i+x_j)+r_{i\max}+r_{j\max}-r_{(i+j)\max} \\ =\ & 2\left[f(x_i)-(\Delta f(x_j)+\cdots+\Delta f(x_i+x_j-1))\right] \\ & -i\cdot\Delta f(x_i)-j\cdot\Delta f(x_j)-(i+j)\cdot\Delta f(x_i+x_j) \\ =\ & 2\left[r_{i\max}+i\cdot\Delta f(x_i)-(\Delta f(x_j)+\cdots+\Delta f(x_i+x_j-1))\right] \\ & -i\cdot\Delta f(x_i)-j\cdot\Delta f(x_j)+(i+j)\cdot\Delta f(x_i+x_j) \\ <\ & i^2\cdot\left|\tilde{\Delta} f(x_i)\right| + i\cdot(\Delta f(x_i)-\Delta f(x_j)) \\ & -(j+1)\cdot(\Delta f(x_j)-\Delta f(x_i+x_j)) \\ & +\left[(\Delta f(x_j)-\Delta f(x_j+1))+\cdots\right. \\ & \left.+(\Delta f(x_j)-\Delta f(x_j+x_i-1))\right] \\ & -\left[(\Delta f(x_j+1)-\Delta f(x_i+x_j))+\cdots\right. \\ & \left.+(\Delta f(x_i+x_j-1)-\Delta f(x_i+x_j))\right] \\ \le\ & i^2\cdot\left|\tilde{\Delta} f(x_i)\right| + i\cdot(j-i)\cdot\left|\tilde{\Delta} f(x_j)\right| \\ & -(j+1)\cdot i\cdot\left|\tilde{\Delta} f(x_j)\right| \\ & +(n-1)\cdot\left(\left|\tilde{\Delta} f(x_j)\right| - \left|\tilde{\Delta} f(x_j+x_i-1)\right|\right) \\ <\ & (-i)\cdot\left|\tilde{\Delta} f(x_i)\right| \\ <\ & 0 \end{aligned} \tag{18}$$

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

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

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

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

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

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

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

Setup: $\mathcal{B}$ generates the private key $k, k_{a1}, k_{a2} \in \mathbb{Z}_p$, and sends the public key $g, g^k, g^{k_{a1}}, g^{k_{a2}}$ to $\mathcal{A}$.

Phase 1: $\mathcal{A}$ adaptively queries the following oracle for polynomial times.

$\mathcal{O}_1$: the challenger $\mathcal{B}$ maintains a $\mathcal{O}_1$-list, which is initially empty. Each entry of $\mathcal{O}_1$-list is $< w, T_w >$. $\mathcal{A}$ can query $\mathcal{O}_1$-list for a keyword w, if w is already in $\mathcal{O}_1$-list, then $\mathcal{B}$ returns T_w to $\mathcal{A}$, otherwise, $\mathcal{B}$ generates the trapdoor T_w for w, adds $< w, T_w >$ to $\mathcal{O}_1$-list, and returns T_w to $\mathcal{A}$.

Challenge: $\mathcal{B}$ chooses a keyword w^* from the keyword dictionary uniformly at random, and returns the encrypted keyword $\hat{w}^* = \left(g^{k\cdot r\cdot H(w*)\cdot k_{a1}} \cdot g^{k_{a1}\cdot k_{a2}}, g^{k\cdot r}\right)$, and trapdoor $T_{w^*} = (g^{H_s(w^*)\cdot r}, g^r)$ to $\mathcal{A}$.

Guess: $\mathcal{A}$ outputs its guess w' for w^*, and sends w' to challenger $\mathcal{B}$. $\mathcal{B}$ returns the encrypted keyword $\hat{w}'$ to $\mathcal{A}$. If $\hat{w}'$ matches T_{w^*}, then $\mathcal{A}$ wins the game.

To complete the proof of Theorem 3, we now compute $\mathcal{A}$'s probability in winning the keyword secrecy game. Assume $\mathcal{A}$ 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 ϵ, therefore, the probability that $\mathcal{A}$ wins the keyword secrecy game is $\frac{1}{u-t} + \epsilon$. □