Thus, FHE concerns an encryption algorithm $E$. The proposed algorithm is based on modular arithmetic in the form $i=1,...,k$. We compute a ciphertext $C$ and are required to be inevitable in $\mathbb{Z}_m$. Secret vectors $s_i$ are randomly chosen elements. Key Generation is the process of generating the keys.

**PROPOSED CRYPTOSYSTEM**

**Basics** FHE supports arbitrary computation over ciphertexts with no need to decrypt and perform computations over original data. Thus, FHE concerns an encryption algorithm $E$ and a decryption algorithm $D$, such that $C_1 = E(X_1), C_2 = E(X_2)$ and $D(f(C_1, C_2)) = f(X_1, X_2)$, where $C_1$ and $C_2$ are ciphertexts, $X_1$ and $X_2$ are plaintexts, $f$ - arbitrary function.

**Key Generation** The main components of the secret key are modulus $M$, vector $m$ of $k$ relatively prime moduli, the set of vectors $s_i, \forall i = 1, \cdots, k$, permutation matrix $P_{C}$, the number $r$ of vectors of randomly chosen elements. Secret vectors $s_i$ are chosen arbitrarily and are required to be inevitable in $\mathbb{Z}_m, j = 1, \cdots, l$.

**Encryption** The inputs are original message $X$, set of multiplication rules. $X$ is represented as a vector $(x_1, x_2, \cdots, x_l)$, s.t. $X = \sum_{i=1}^{l} x_i \ mod \ M$. Then the secret key vectors $(s_1, \cdots, s_k)$ is applied to compute a ciphertext $C = (c_1, c_2, \cdots, c_k)$ as $c_i = s_i \cdot X \ (mod \ (m_i))$, for $i=1,\cdots,k$.

**Decryption** To restore the ciphertext from permutation we apply $P_{C}$ matrix first: $C = P_{C}C$. Then, apply the inverses $s_i^{-1}$ of the secret vectors $s_i$ for decryption and use Chinese remainder theorem [1] to find $X$, that solve the system of equations of type $\hat{X} = (c_i \cdot s_i^{-1}) \ mod \ m_i$.

**Multiplication** The multiplication of two ciphertexts leads to the increase of the result’s size about 4 times. To solve this problem we first introduce a set of vectors $(\zeta_{ij}, \zeta_{ij}^*)$ with bases $(\zeta_{ij}^b)$ to represent entries of ciphertext $c_{ij}$ and $c_{ij}^*$ as products $(\zeta_{ij} \cdot \zeta_{ij})$. Then the public key sent to the server $\gamma$, is estimated as $\gamma_{ij} = (\zeta_{ia}^b \cdot S_1^{-1}(mod_m)) \cdot (\zeta_{ib}^b \cdot S_1^{-1}(mod_m))$.

**CONCLUSION**

In this research, we propose the FHE scheme which is well suited for the efficient implementation on the computer. The use of modular arithmetic prevents overflow involving legitimate computation range. Multiplication tables address the problem of exponential data growth and allow to work with rational numbers, thus increasing the strength of the encryption scheme. Domingo-Ferrer’s FHE scheme is turned out to be the special case of the scheme proposed in this paper. Both schemes involve random splitting of the original number into small secret values $\in \mathbb{Z}_m$. However, instead of choosing a single modulus $m$ and a vector $\bar{s}$ of invertible values as a secret key, our scheme uses the secret vector of $k$ moduli $m_i$ and a set of $k$ secret vectors $s_j$ (invertible in $\mathbb{Z}_{m_i}$). Thus, our scheme is more secure as it requires a number to be represented as a matrix of values in the rings with the different bases $m_i$. Our scheme generalizes Domingo-Ferrers’ scheme to multivariable functions and extends it to encompass the application of multiplication operations over encrypted data without the growth of the result vector’s length.

As a future work, we are planning to integrate RSA public-key cryptosystem with our FHE scheme to enhance security features in RSA for the cloud-based applications. Also, we intent to work in the direction of adapting our algorithm for genomic data encryption, taking into considerations results of [6].

**REFERENCES**