EVAULATION OF HOMOMORPHIC ENCRYPTION IMPLEMENTATION ON IOT DEVICE

Goran Đorđević[1], Milan Marković[2], Pavle Vuletić[3]

[1]*AET Europe, Arnhem, The Netherlands, School of Electrical Engineering, Belgrade, Serbia, goran.djordjevic@aeteurope.com*

[2]Pan-European University Apeiron, Banja Luka, Serbian Republic, BiH, *milan.z.markovic@apeiron-edu.eu*

[3]School of Electrical Engineering, Belgrade, Serbia, *pavle.vuletic@etf.bg.ac.rs*

Contribution to the State of the Art

https://doi.org/10.7251/JIT2201032DJ UDC: 004.738.5:621.391

Abstract: An encryption scheme is homomorphic if it supports operations on encrypted data. Homomorphic encryption allows a device to perform arbitrary computations on encrypted data without user secret key. Recently it is introduced new homomorphic encryption schemes with improved performance that can be implemented in IoT device in production environments. The IoT concept encompasses devices, sensors, and services existing within an interconnected infrastructure with an efficient access to sample computational facilities. In this paper we evaluated features of exact arithmetic homomorphic encryption mechanisms: BFV and BGV and approximate homomorphic encryption scheme: CKKS. In the paper we measured performances of operations of homomorphic encryption schemes: BGV, BFV and CKKS that are implemented in Raspberry Pi 4 IoT device.

Keywords: Homomorphic encryption, Raspberry Pi IoT device, HE schemes performance.

INTRODUCTION

Traditional encryption schemes, both symmetric and asymmetric, were not designed to perform computations on the ciphertext in a way that would pass through the encryption to the underlying plaintext without using the secret key. The property would in many contexts be considered a vulnerability.

Homomorphic encryption differs from basic encryption methods in that it allows computation to be performed directly on encrypted data without requiring access to a secret key [1]. Homomorphic encryption schemes can be taken as a generalization of public key encryption mechanisms.

For long time the main issue of homomorphic encryption implementation was its low performance. Due to the issue the homomorphic encryption could not applied in production environments. However, in last few years it is introduced new homomorphic schemes with improved performance that make them suitable appliance in production systems. Additionally, it is proposed new optimization mechanisms of existing homomorphic operations that improved them performance and they can be used in production systems.

Fully Homomorphic Encryption (FHE) scheme is cryptographic mechanism that support arbitrary level of computations on ciphertext (addition, rotation, multiplication) not knowing the value of its secret key. Modern fully homomorphic encryption algorithms use complex mechanims on lattice structures and Ring-LWE (Ring Learning With Errors, RLWE) scheme [2].

According to latest researchs modern homomorphic encryption schemes are resistant to quantum computer attacks. There are no known mechanisms that would use the features of quantum computers to break homomorphic algorithms in polynomial time.

An example of cloud-based scenario with deployed homomorphic encryption operations is shown in Figure 1.

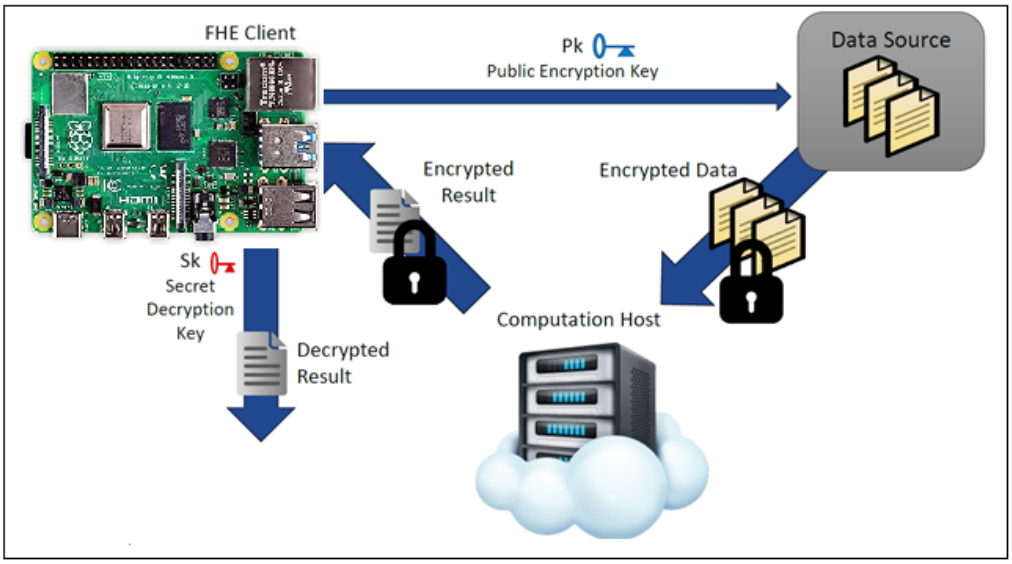

Figure 1: *Client-server homomorphic encryption scenario*

At server side data is processed on the server in the encrypted form. The results are remained in the encrypted form and they are sent back to the device who can decrypt the data and use the result.

In the work homomorphic encryption mechanisms are implemented in Raspberry Pi 4 Model B. The first generation of Raspberry (Pi 1) was released in the year 2012, that has two types of models namely model A and model B.

In this work we are estimated the features and performance of exact (BGV and BFV) and approximate (CKKS) homomorphic encryption schemes are discussing the conveniences and constraints of these schemes for use in the cloud-based systems. Performance assessment of the FHE schemes implemented in IoT device has not been much explored in the literature. The work will provide a better insight into the current state of the work on homomorphic encryption and its suitability for deployments in real IoT based systems.

The CKKS homomorphic scheme enables approximate homomorphic encryption for efficient processing of floating-point data. Using CKKS homomorphic scheme can be considered as efficient choice for homomorphic logistic regression and other machine learning tasks.

The paper is organized as follows. Section 'Related work' gives an overview of the related work in the field of the performance evaluation of the FHE schemes. The most important properties of homomorphic encryption and the classification of the homomorphic encryption schemes are presented in Section 'Properties of homomorphic encryption'. An application of Homomorphic Encryption: IoT (Internet of Things) case is shown in Section 'An example of appliance of homomorphic encryption'. The main features and description of modern HE schemes: BGV [3], BVF [4] and CKKS [5] are elaborated in Section 'Homomorphic schemes'. In Section ‚Experimental analysis' is shown results of experimental analysis. Conclusions are given in Section 'Conclusions'.

Related work

In article [6] is described only BGV Encryption Scheme for IoT Systems. In the work is not shown homomorphic encryption operations performance for other relevant modern schemes like CKKS and BFV.

In article [7] is analyzed and described test results only for CKKS approximate homomorphic encryption scheme. In the work is not described neither shown homomorphic encryption operations performance for other schemes like BGV and BFV.

In article [8] is described Fast Number Theoretic Transform operations for Ring-LWE on 8-bit AVR Embedded Processor but in the article is not shown performance of homomorphic encryptions schemes like CKKS, BFV and BGV.

Unlike the previous work, in this work we give experimental results for exact (BGV, BFV) and approximate (CKKS) homomorphic encryption schemes that are implemented in IoT device.

PROPERTIES OF HOMOMORPHIC ENCRYPTION

There are four main types of homomorphic schemes [9]:

- Partially Homomorphic Encryption (PHE). The partially homomorhpic encryption supports any number of operations (multiplicatio-on, addition), but it is limited to just one type of operation.
- Somewhat Homomorphic Encryption (SHE) allows both addition and multiplication, but it can perform a limited number of operations.
- Levelled Homomorphic Encryptions (LHE). This scheme can evaluate only functions with certain complexity (depth).
- Fully Homomorphic Encryption. The scheme allows any number of addition or multiplication operations. An Fully Homomorphic Encryption scheme can evaluate unbounded complexity (depth).

Homomorphic Encryption should support two main homomorphic operations [10]:

- Additive Homomorphic Encryption (Figure 2).
- Multiplicative Homomorphic Encryption.

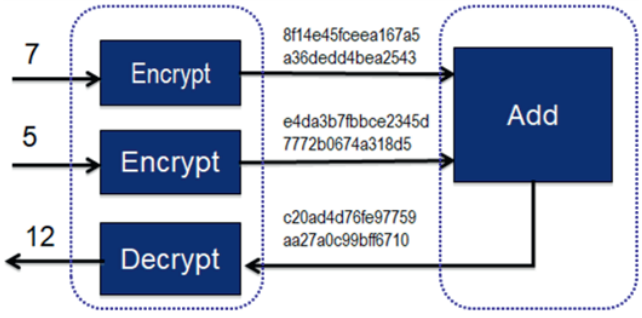

Figure 2: *An example of Additive Homomorphic Encryption*

Homomorphic encryption is additive, if [11]: Enc $(m_1 + m_2)$ = Enc (m_1) + Enc (m_2); $\forall m_1, m_2 \in M$.

Homomorphic encryption is multiplicative, if [11]: Enc $(m_1 * m_2)$ = Enc (m_1) * Enc (m_2); $\forall m_1, m_2 \in M$.

The most popular classes of homomorphic schemes, given with their main properties (Figure 3), are:

- Boolean circuit (Fastest Homomorphic Encryption in the West (FHEW) [12] and Fast Fully Homomorphic Encryption over the Torus (TFHE) [13]):
  - Plaintext data are coded as bits;
  - Computations are performed by using Boolean circuits.
- Modular integer arithmetic (BGV, BFV):
  - Plaintext data are coded as integer modulo a plaintext;
  - Computations are expressed as integer modulo arithmetic.
- Approximate number arithmetic (CKKS):
  - Plaintext data are coded as real (or complex) numbers;
  - Computations are performed in a way similar to floating-point arithmetic but dealing with fixed-point numbers.

Modern homomorphic encryption schemes are based on usage of lattice cryptography with errors LWE [14]. Lattices have an important role in modern cryptography, especially in the context of the research on post-quantum cryptography. So far it is not reported in the literature fact that is claimed

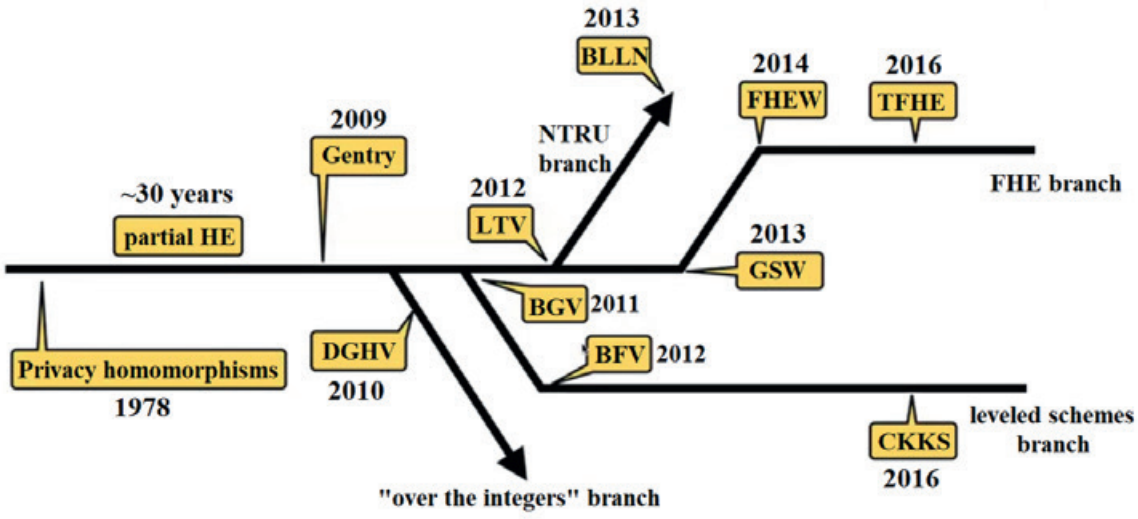

Figure 3: *History of Homomorphic scheme*

that it can break lattice-based cryptographic algorithms using quantum computer algorithms.

The modern homomorphic encryption schemes apply structured lattices i.e. they use Ring-LWE mechanism [2]. The Ring-LWE reduces both important factors: computation time and key length.

The Ring-LWE implementation is based on power-of-two cyclotomic rings:

$$R_q = \exists_q / \langle x^n + 1 \rangle$$

The optimized Residue Number System (RNS) variants show significant increase performance compared to their previous respective implementations [15]. The Residue Number System works with native (machine-word size) integers because it is faster than multi-precision integer arithmetic. The Residue Number System breaks rings of large bit-width integers into a parallel set of rings (<64-bit residues) allowing performant computation on 32/64-bit CPU architecture.

Modulus q is represented as product of integers:

$$q = \prod_{i=1}^{k} q_i$$

Modulus q is a functional parameter which determines how many computations are allowed without the application of bootstrapping procedure [16].

One of the most important feature of the homomorphic encryption mechanisms is that they add noise to a ciphertext during performing encryption, multiplication, addition and rotation homomorphic operations. Homomorphic operations, especially multiplication, increase level of the noise. If the noise becomes too large ciphertext can not be decrypted successfully. Noise budget is the total amount of noise that can be added until the decryption fails [17]. The bootstrapping is the procedure of "refreshing" a ciphertext by running homomorphically decryption operation that reduces level of noise.

All analysed homomorphic encryption schemes (BGV, BFV, CKKS) support the following homomorphic operations [18]:

- Addition;
- Multiplication;
- Rotation.

An example of appliance of homomorhic encryption

The data privacy concerns are increasingly affecting the Internet of Things (IoT) in which it is very challenging to protect the privacy of the underlying data [7]. Functional architecture of IoT platform, its core decription and an example of its application can be found in [19]. An overview of secure model of SOA based healtcare systems with mobile web service is shown in [20]. In the model data the medical data (DNA, patient data) can be protected using homomorphic encryption mechanisms.

It can be detected following classes of attacks on systems based on IoT devices and cloud servers [7]:

- Network attacks. A network attacker sees all traffic across all non-private networks. A network attacker may act actively or passively and may exploit side-channel leakage transmitted over the network to learn user data.
- Cloud attacks. A cloud attacker may refer either to an external entity who can corrupt a cloud sever or a cloud infrastructure provider themselves. A cloud attacker may attempt to bypass traditional protections in the cloud to obtain read access to private data through conventional software methods (e.g. buffer overflows), side-channel attacks such as timing or power analysis, or physical attacks.
- Device attacks. The attack refers to an attack on the device itself. These attacks may be remote and injected through direct or indirect software attacks.

Improvements and optimization of homomorphic encryption operations can make homomorphic encryption schemes enough efficient for practical usage in real systems.

Encryption is asymmetric, while evaluation refers to computation on encrypted data (performed by cloud based server) to provide encrypted results. Finally, decryption can be performed either by a trusted party (Smartphone, PC) or by same IoT device (Figure 4).

The before mentioned homomorphic encryption schemes support a technique called batching. The batching technique refers to the encoding of multiple messages into a single plaintext. The resulting batched plaintext can be encrypted into a single corresponding batched ciphertext.

Moreover, computation on batched ciphertexts can be performed in a Single Instruction Multiple Data (SIMD) way on the underlying messages, reducing the cost of homomorphic evaluation by several orders of magnitude.

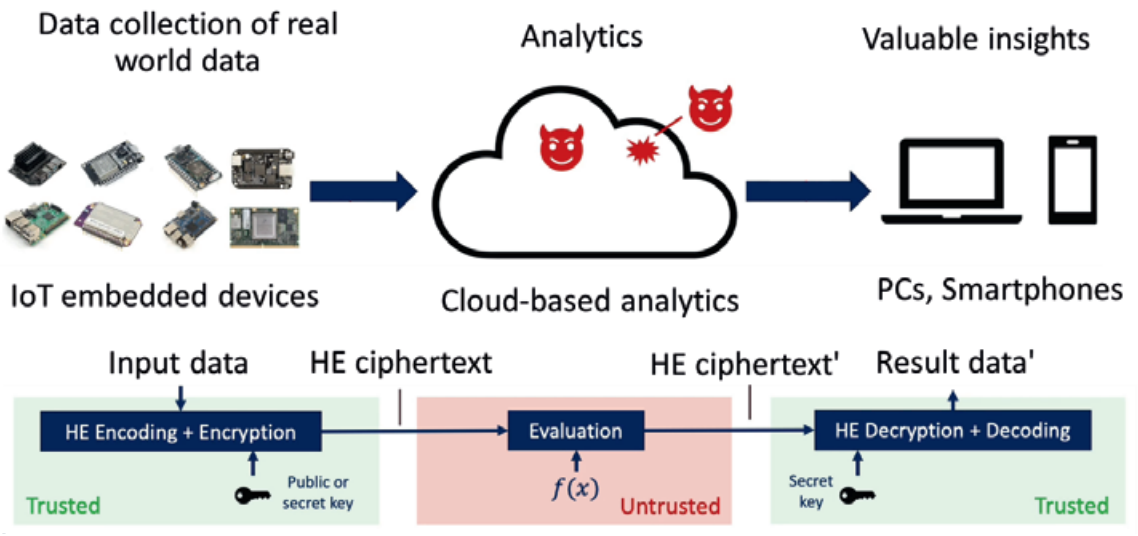

Figure 4: *Appliance of HE with IoT devices*

HOMOMORPHIC SCHEMES

The homomorphic encryption scheme named BGV was proposed in [3]. The BGV is a levelled homomorphic encryption scheme, meaning that the parameters of the scheme depend on the multiplicative depth that the scheme is capable to evaluate. Multiplicative depth of BGV scheme determines how many sequential multiplications can be performed.

The BFV scheme, presented in [4] is a homomorphic cryptographic scheme based on the Ring-LWE problem in a cryptographic lattice.

The homomorphic encryption scheme named CKKS, presented in [5] is known as Homomorphic Encryption for Arithmetic of Approximate Numbers (HEAAN). Operations supported in CKKS homomorphic encryption scheme are shown in Figure 5. The

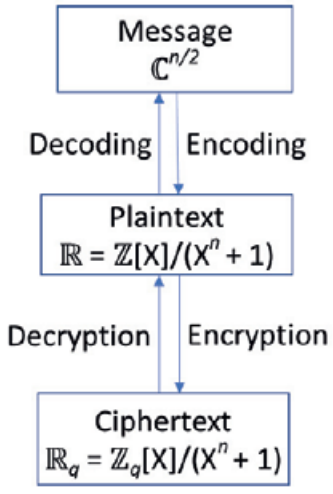

Figure 5: *Homomorphic encryption operations in CKKS*

homomorphic encryption CKKS scheme enables computations on vectors of complex values.

The approximate homomorphic encryption scheme CKKS has the following characteristics:

- $Dec(Enc(m)) \approx m$;
- $Dec(ct_1 * ct_2) \approx Dec(ct_1) * Dec(ct_2)$;
- Noise bounds are determined by the parameter set.

In the CKKS scheme noise is considered as a part of numerical error in approximate computation. It supports homomorphic rounding-off.

EXPERIMENTAL ANALYSIS

In the experimental analysis we measure the time needed for performing the following homomorphic operations: Public key encryption (Table 1), Secret key decryption (Table 2), Homomorphic addition (Table 3), Homomorphic multiplication without relinearization (Table 4) and Homomorphic rotation (Table 5). We evaluated the use of the following homomorphic mechanisms:

- BGV,
- BVF, and
- CKKS;

which are implemented in the following open-source libraries respectively:

- Microsoft SEAL [21] and
- Palisade [16];

Palisade [16] is multi-threaded library written in C++ 11. It uses the Number Theory Library (NTL) [22] to accelerate underlying mathematical operations. Palisade supports more schemes, including

BFV, BGV, CKKS. Additionally, Palisade supports multi-party extensions of certain schemes and other cryptographic primitives like Proxy Re-Encryption (PRE) and digital signatures [9].

Microsoft Simple Encrypted Arithmetic Library (SEAL) [21] is a homomorphic encryption library that allows addition and multiplication operations on encrypted integers or real numbers. Microsoft SEAL is written in C++11 and contains a .NET wrapper library for the public API. The latest available version is 4.0.0.

The homomorphic encryption code was performed on a Raspberry Pi 4 model B IoT device with:

- Broadcom BCM2711 quad-core Cortex-A72 (ARM v8) 64-bit SoC @ 1.8GHz;
- 4GB LPDDR4-3200;
- Bluetooth 5.0, Bluetooth Low Energy (BLE);
- Debian GNU/Linux 11 (bullseye);
- Hardware model: BCM2835.

All homomorphic encryption operations are performed with homomorphic encryption ciphertext dimension n = 8192.

In tables 1, 2, 3, 4, and 5 show the results of public key encryption, secret key decryption, homomorphic encryption addition, homomorphic encryption multiplication, and homomorphic encryption rotation tests respectively, where:

- Times in the last two columns (HE Library) are expressed in microsecond (µs);
- Each homomorphic encryption related operation performed 1000 times;
- We used 128-bit homomorphic encryption security level. The homomorphic encryption standard set with more than 128 bits of security with reference to classical cryptography computer attacks.
- Ciphertext dimension is *n.*

The public key encryption operation in all tested homomorphic encryption schemes BFV scheme has the best performance when the Palisade library is used.

Table 1: *Public key encryption in IoT device*

HE schemes	HE parameter	HE library	
	Ciphertext dimension n	Palisade	SEAL
BFV	8,192	7,820 µs	14,250 µs
BGV	8,192	8,511 µs	14,189 µs
CKKS	8,192	8,209 µs	16,888 µs

In Raspberry Pi 4 IoT device the public key encryption (Table 1) is achieved faster performance using Palisade library than SEAL library.. In Palisade the public key encryption the best performance is achieved using BFV homomorphic encryption scheme. In SEAL the public key encryption the best performance is achieved using BGV homomorphic encryption scheme.

Table 2: *Secret key decryption in IoT device*

HE schemes	HE parameter	HE library	
	Ciphertext dimension n	Palisade	SEAL
BFV	8,192	1,607 µs	5,285 µs
BGV	8,192	2,094 µs	5,089 µs
CKKS	8,192	11,644 µs	889 µs

The secret key decryption (Table 2) in CKKS scheme has better performance in the SEAL than in the Palisade library.

The secret key decryption of exact homomorphic encryption schemes (BGV, BFV) faster performance is achieved in Palisade than in SEAL library. In Palisade library secret key decryption is fastest by using BFV homomorphic encryption scheme and in SEAL the operation is fastest by using CKKS approcimate homomorhic encryption scheme.

Table 3: *Homomophic encryption addition in IoT device*

HE schemes	HE parameter	HE library	
	Ciphertext dimension n	Palisade	SEAL
BFV	8,192	358 µs	1,280 µs
BGV	8,192	493 µs	1,341 µs
CKKS	8,192	808 µs	1,632 µs

In Raspberry Pi 4 IoT device the homomophic encryption addition (Table 3) is achieved faster performance using Palisade library than SEAL library. In both library the best performance of homomophic encryption addition is achieved using BFV scheme.

Table 4: *Homomophic encryption multiplication in IoT device*

HE schemes	HE parameter	HE library	
	Ciphertext dimension n	Palisade	SEAL
BFV	8,192	17,845 µs	57,332 µs
BGV	8,192	1,317 µs	16,605 µs
CKKS	8,192	1,447 µs	3,293 µs

In Raspberry Pi 4 IoT device the homomophic encryption multiplication (Table 4) is achieved faster performance using Palisade library than SEAL library. In Palisade the homomophic encryption multiplication is achieved the best performanse using BGV scheme. In SEAL the homomophic encryption multiplication is achieved the best performanse using CKKS scheme.

Table 5: *Homomophic encryption rotation in IoT device*

HE schemes	HE parameter	HE library	
	Ciphertext dimension n	Palisade	SEAL
BFV	8,192	3,940	16,207
BGV	8,192	3,641	17,482
CKKS	8,192	9,307	16,288

In Raspberry Pi 4 IoT device the homomophic encryption rotation (Table 5) is achieved faster performance using Palisade library than SEAL library. In Palisade the homomophic encryption rotation is achieved the best performanse using BGV scheme. In SEAL the homomophic encryption rotation is achieved the best performanse using BFV scheme that is slightly faster than in case of appliance the operation using CKKS.

CONCLUSIONS

Homomorphic encryption enables performing computations on the encrypted data, without decrypting them. The work compares the time needed to execute homomorphic operations, like, public key encryption, secret key decryption, addition, multiplication, and rotation implemented in the open-source libraries: Microsoft SEAL and Palisade. The operations are compared for BGV, BFV and CKKS homomorphic encryption schemes implemented in the libraries.

Homomorphic operations that are usually performed at client side: public key encryption and secret key decryption in case of appliance exact arithmetic homomorphic operations are achieved best performance using Palisade library. In case of usage of approximate arithmetic operations (CKKS) public key encryption operations have better performance in Palisade than in SEAL library. The SEAL library provides better performance in secret key decryption operation.

To demonstrate the possibility of implementation of homomorphic encryption related operations in IoT device we deployed the different homomorphic encryption schemes on Raspberry Pi 4 model B based IoT platform. Our results show that homomorphic encryption operations can be applied on embedded devices.

The CKKS homomorphic encryption scheme can be considered as natural choice for IoT devices since it can efficiently perform secure computation on the type of real-valued data often sampled by sensors. In this sense in future work, it can be considered method of efficient implementation of CKKS scheme especially operations of public key encryption and private key description that are performed on client (IoT) side.

The perfdević of current fully homomorphic encryption schemes, especially for large parameters, can still be improved. Further improvement can be achieved by implementation low-level homomorphic operations in an assembly language which is executed on Raspberry Pi 4 IoT hardware platform.

REFERENCES

[1] A. Acar, H. Aksu, A. Selcuk, and M. Conti, "A Survey on Homomorphic Encryption Schemes: Theory and Implementation," ACM Comput. Surv. 1, 1, Article 1, http://dx.doi.org/10.1145/3214303, 2018.

[2] V. Lyubashevsky, C. Peikert, and O. Regev, "On ideal lattices and learning with errors over rings," Journal of the ACM (JACM) 60, no. 6, 2013.

[3] Z. Brakerski, C. Gentry, and V. Vaikuntanathan, "Fully Homomorphic Encryption without Bootstrapping," Cryptology ePrint Archive, Report 2011/277. https://eprint.iacr.org/2011/277, 2011.

[4] J. Fan and F. Vercauteren, "Somewhat practical fully homomorphic encryption," IACR Cryptology ePrint Archive, 2012:144, 2012.

[5] J.H. Cheon, A. Kim, M. Kim, and Y. Song, „Homomorphic encryption for arithmetic of approximate numbers," Cryptology ePrint Archive, Report 2016/421, https://eprint.iacr.org/2016/421, 2016.

[6] W. Yuan, H. Gao, "An Efficient BGV-type Encryption Scheme for IoT Systems," https://doi.org/10.3390/app10175732, 2020.

[7] D. Natarajan, W.Dai, "SEAL-Embedded: A Homomorphic Encryption Library for the Internet of Things", IACR Transactions on Cryptographic Hardware and Embedded Systems, Vol. 2021, pp 756-779.

[8] H.Seo, H.Kwon, Y.Kwon, K.Kim, S.Choi,H.Kim, K.Jang, "Fast Number Theoretic Transform for Ring-LWE on 8-bit AVR Embedded Processor," Sensors (Basel). 2020 Apr 5;20(7):2039. doi: 10.3390/s20072039.

[9] G. Đorđević, M. Marković, P.Vuletić, "Performance comparison of homomorphic encryption scheme implementations", 8th International Conference on Electrical, Electronic and Computing Engineering, IcETRAN 2021, pp 514-519.

[10] C. Aguilar Melchor, M. Kilijian, C. Lefebvre, T. Ricosset, "A Comparison of the Homomorphic Encryption Libraries HElib, SEAL and FV-NFLlib," in: Lanet JL., Toma C. (eds)

Innovative Security Solutions for Information Technology and Communications. SECITC 2018. Lecture Notes in Computer Science, vol 11359. Springer, Cham. https://doi.org/10.1007/978-3-030-12942-2_32, 2019.
[11] T. Maha, S. Hajji, and A. Ghazi, "Homomorphic encryption applied to the cloud computing security," in Proceedings of the World Congress Engineering, vol. 1, pp. 4-6, 2012.
[12] L. Ducas and D. Micciancio, "FHEW: bootstrapping homomorphic encryption in less than a second," in E. Oswald and M. Fischlin, editors, Advances in Cryptology – EUROCRYPT 2015 - 34th Annual International Conference on the Theory and Applications of Cryptographic Techniques, So_a, Bulgaria, April 26-30, 2015, Proceedings, Part I, volume 9056 of Lecture Notes in Computer Science, pages 617-640. Springer, 2015.
[13] I. Chillotti, N. Gama, M. Georgieva, and M. Izabachene, "Faster packed homomorphic operations and e_cient circuit bootstrapping for tfhe," in Advances in Cryptology-ASIACRYPT 2017: 23rd International Conference on the Theory and Application of Cryptology and Information Security, pages 377-408. Springer, 2017.
[14] O. Regev, "The learning with errors problem," in Blavatnik School of Computer Science, Tel Aviv University Invited survey in CCC, 2010.
[15] J. H. Cheon, K. Han, A. Kim, M. Kim, and Y. Song, "A full rns variant of approximate homomorphic encryption," Cryptology ePrint Archive,Report 2018/931, https://eprint.iacr.org/2018/931, 2018.
[16] Y. Polyakov, K. Rohloff, G.W. Ryan, and D. Cousins, "PALISADE Lattice Cryptography Library User Manual (v1.10.6)", 2020.
[17] S. Sathya, P. Vepakomma, R. Raskar, R. Ramachandra, and S. Bhattacharya, "A Review of Homomorphic Encryption Libraries for Secure Computation," http://arxiv.org/abs/1812.024, 2018.
[18] M. Albrecht, M. Chase, H. Chen and others, "Homomorphic encryption standardization," homomorphicencryption.org, 2018.
[19] G. Đukanović, G. Popović, D. Kanellopoulos, "Scaling complexity comparison of an ACO-based routing algorithm used as an IoT network core," Journal of Information Technology and Applications, JITA 10(2020) 2:73-80, (UDC: 004.738.5:004.738.057.4), (DOI: 10.7251/JIT-2002073DJ), Volume 10, Number 2, Banja Luka, December 2020 (69-128), ISSN 2232-9625 (print), ISSN 2233-0194 (online),
[20] G. Đorđević, M. Marković, "On Possible Cryptographic Optimization of Mobile Healthcare Application," JITA – Journal of InformationTechnology and Applications Banja Luka, JITA 9(2019) 2:80-88, (UDC: 004.056.55:621.39), (DOI: 10.7251/JIT1902080DJ), Volume 9, Number 2, Banja Luka, december 2019.(49-128), ISSN 2232-9625 (print), ISSN 2233-0194 (online)
[21] K. Laine, "Simple Encrypted Arithmetic Library 2.3.1," 2017.
[22] V. Shoup and others, "NTL: A library for doing number theory," http://www.shoup.net/ntl.

Received: January 17, 2022.
Accepted: May 24, 2022.

About the authors

Goran V. Đorđević was born 1972 in Novi Sad, Serbia. He received a BSc in Computer Science at the Technical Military Academy in 1996. Afterwards he did his post-graduate studies, at the Faculty of Electrical Engineering of University of Belgrade where he received a MSc. Currently employed as a senior software developer in AET Europe. His main areas of interest are smart card security and smart card applications, security protocol design, mobile devices, tokens, Internet of Things and information security.

Milan Marković received B.S.E.E., M.S.E.E., and Ph.D. degrees in electrical engineering from Faculty of Electrical Engineering, University of Belgrade, Serbia, in 1989, 1992, and 2001, respectively. He is an Associate Professor on College of Information Technology, Pan-European University of Apeiron in domain of information security courses. His research interests are mainly in public key infrastructure, information security, cryptographic algorithms, mobile security, identity management, secure e/m-banking and e/m-government, trust services, ISMS, Blockchain, etc. He has published more than 320 scientific papers.

Pavle Vuletić obtained his BSc, MSc and PhD in Computer Systems and Network Architecture from University of Belgrade, School of Electrical Engineering (ETF). He used to work on all positions from network engineer to the deputy director of AMRES, national research and education network where he participated in the establishment of the first national CSIRT team. He is currently the associate professor at the ETF at the Department of Computer Engineering and Information Theory, teaching Data Security, Computer Systems and Network Security, Advanced Computer Networks and Software Defined Networks courses and the head of the Laboratory for Information Security at the ETF. His research interests span from network and systems security, intrusion detection, network and system performance and monitoring to software defined networks and network and systems management.

For citation

Goran Đorđević, Milan Marković, Pavle Vuletić, Evaulation of Homomorphic Encryption Implementation on Iot Device, *JITA – Journal of Information Technology and Applications, Banja Luka*, Pan-Europien University APEIRON, Banja Luka, Republika Srpska, Bosna i Hercegovina, JITA 12(2022) 1:32-39, (UDC: 004.738.5:621.391), (DOI: 10.7251/JIT2201032DJ), Volume 12, Number 1, Banja Luka, June (1-64), ISSN 2232-9625 (print), ISSN 2233-0194 (online), UDC 004