# Assessment of Cloud-based Health Monitoring using Homomorphic Encryption

Ovunc Kocabas*, Tolga Soyata*, Jean-Philippe Couderc*†, Mehmet Aktas†, Jean Xia*†, Michael Huang*
*Dept. of Electrical and Computer Engineering
University of Rochester
Rochester, NY 14627
{ovunc.kocabas,tolga.soyata,micheal.huang}@rochester.edu
†URMC Medical Center
Rochester, NY 14627
mehmet_aktas@urmc.rochester.edu
{jean-philippe.couderc,jean.xia}@heart.rochester.edu

***Abstract*—Current financial and regulatory pressure has provided strong incentives to institute better disease prevention, improved patient monitoring, and push U.S. healthcare into the digital era. This transition requires that data privacy be ensured for digital health data in three distinct phases: I. acquisition, II. storage, and III. computation. Each phase comes with unique challenges in terms of proper implementation and privacy.**

**While the privacy of the data can be ensured with existing AES encryption techniques in phases I (acquisition) and II (storage), to enable healthcare organizations to take advantage of cloud computing using resources such as Amazon Web Services, phase III (computation) must also enable the privacy of the data. Currently, there exists no system to enable direct computation in the cloud while assuring data privacy.**

**Fully Homomorphic Encryption (FHE) is an emerging cryptographic technique to permit computation on encrypted data directly in the cloud without the need to bring the data back to the computational node. However, this promising technique comes with significant performance- and storage-related challenges. While it will take more years before true FHE is mainstream, we provide a feasibility study for its application to a simple long-term patient ECG-data monitoring system.**

## I. Introduction

Accelerated by the US government to modernize the US health system, cloud computing based medical applications are an active research area [1]. While one motivation for this is to reduce operational costs at the healthcare organization (HCO) by eliminating the datacenters managed by the HCO, an equally important motivation is to improve healthcare by providing the doctors with long-term patient data as an auxiliary diagnosis tool. We define *long term data* as digitized measurements of a patient's vitals over the course of his/her treatment (e.g., blood pressure, ECG) that is longer than what can be obtained within the HCO (e.g., multiple days or weeks). In this paper, we study the possibility of a system to provide long-term health monitoring strictly by utilizing cloud computing resources (e.g., Amazon EC2 [2], Microsoft Azure [3], or Google [4]), accessed by *thin* devices such as tablets or mobile phones, while assuring complete patient data privacy.

To turn this vision into reality, the privacy of the medical data acquired outside the HCO must be ensured at three distinct phases of its transition: **Phase I. Acquisition**, where the medical data are acquired outside the HCO via thin disposable devices such as ECG patches [5], **Phase II. Storage**, where the data is stored permanently in the cloud for future use, and, **Phase III. Computation**, where the data is processed. While existing AES-based encryption techniques [6] can ensure data privacy in phases I and II, achieving computation in the cloud (i.e., during Phase III) requires operating on encrypted data. In this paper, we conduct a feasibility study of achieving Phase III strictly in the cloud by using the emerging Fully Homomorphic Encryption (FHE) techniques [7], [8] on a restricted set of remote cardiac health monitoring applications by using existing commodity ECG patches [5]. We focus on the case, where the HCO has no resources allocated to the storage or computation (i.e., no datacenter to host the medical application), except the *thin* mobile devices (i.e., devices with limited computational and storage capability), which strictly act as the graphical user interface (GUI) devices.

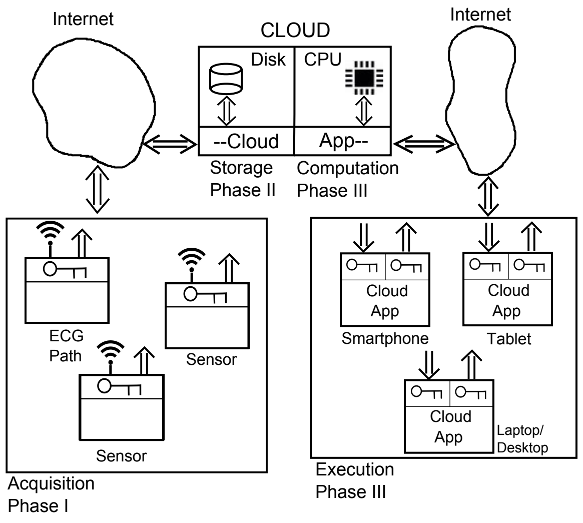

Fig. 1. Proposed system for long term health monitoring through a cloud-based application.

Our proposed system is shown in Figure 1, where phase I (Acquisition) of the long-term health monitoring is achieved via the use of remote sensors that are capable of regular AES encryption and transmission to the cloud via existing wireless access points. Without loss of generality, we specifically focus on the ECG-based applications and the resulting improved diagnosis possibilities for heart diseases. Our system can be applied to any sensor that has similar capabilities with a backend application that has similar characteristics. In this system, phase II (storage) and III (computation) are strictly in the cloud, which leaves only the GUI responsibility to the mobile end-device during the access to the monitoring results.

The goal of our system is to push the entire workload into the cloud, and allow thin devices to be used as acquisition sensors and GUI. This will enable true cloud computing by turning the *end nodes* of this system (i.e., acquisition and GUI) into simple (disposable) devices, while leaving the core of the system in the cloud. Furthermore, since nearly no information is stored on the end-devices, data privacy concerns due to the loss of the ECG patches and/or mobile devices by the patient or the doctors (or nurses) are minimized, if not eliminated.

To the our best knowledge, this is the first paper that focuses on the application of FHE to long-term patient monitoring. Our contributions in this paper are summarized as follows:

1) we propose a cloud-based medical application where the end nodes are simple and potentially disposable. The core of the application resides strictly in the cloud, permitting flexibility to the HCO in its data center strategies;
2) we formulate Full Homomorphic Encryption (FHE) as the core of this idea;
3) we identify the challenges in making this possible for a specific application based on remote-ECG monitoring;
4) we determine what is possible within the following few years while FHE acceleration is being widely researched [9].

The rest of the paper is organized as follows. In Section II, we provide background information on Electrocardiogram (ECG) and Fully Homomorphic Encryption (FHE). In Section III, we introduce our proposed system in detail and identify the challenges of different parts of it. Section IV is where we present our results on existing ECG based patient data derived from the THEW database [10], followed by our conclusions and propositions for future work in Section V.

## II. Background

To gain an insight into the challenges in applying FHE into a broader scope of medical applications, we will provide background information on our pilot medical application based on Electrocardiogram (ECG) and FHE.

### A. Case Study: Computing The Average Heart Rate

The proposed cloud-based technology for health information will be designed to support an exhaustive list of data that are routinely acquired by healthcare organizations (HCO) from their patients such as: echo/MRI imaging data, subject drug treatment, and physiological monitoring signals. In this work, we have opted to limit our feasibility assessment to a simple, yet real, set of data acquired from the THEW worldwide ECG database [10]. This data contain information about the electrical activity of the heart of the patient recorded for 24-hours. In order to demonstrate the feasibility of our concept, we selected average heart rate calculation as our case study. To compute the average heart rate we use information extracted from the annotation file of the ECG. The annotation file provides information related to cardiac contraction for each beat and the temporal distance ($toc$) between consecutive beats. The $toc$ values between consecutive beats will be used to calculate the average heart rate.

### B. Fully Homomorphic Encryption (FHE)

An FHE scheme enables computation of arbitrary functions on encrypted data without compromising the privacy of the original data. In 2009, Gentry [7] proposed the first plausible mechanism for an FHE scheme which could perform an arbitrary number of additions and multiplications homomorphically. In his proposal, random noise is introduced to ciphertext for obstructing the message. The noise inside the ciphertext grows with each addition and multiplication. After a number of operations, the magnitude of the noise exceeds the threshold that can be tolerated during the decryption. Gentry proposed recryption method to overcome the increasing noise, in which the noise in the ciphertext is reduced by evaluating decryption homomorphically. However the cost of performing recryption is prohibitively high which makes proposed FHE scheme impractical, as we will also demonstrate with experimental results. Following Gentry's scheme, number of FHE schemes and implementations have been proposed to date [8], [11]–[20] to make FHE more practical. At present, the BGV scheme [8] and its implementation [19] is the one of the most promising candidates for a practical FHE scheme. We will also demonstrate the results of BGV later.

## III. Components of the Medical Application

The core of the proposed system is based on delegating almost the entire computational workload to the cloud. The system is composed of three distinct parts: 1) Real-time medical data acquisition devices, 2) Cloud-based computation and storage, and 3) GUI end nodes. In the following subsections, these parts will be analysed separately.

### A. Real-time medical data acquisition devices

Acquisition devices are the initiating nodes of real-time medical data. These nodes can be either the disposable ECG patches attached to a patient [5], mobile ECG carts used in hospitals. These devices are assumed to be capable of transmitting the acquired data wirelessly in an AES encrypted format to protect the patient's privacy. While AES [6] is available even in the least expensive embedded devices [21], FHE can only be handled in the cloud due to its computational requirements.

### B. Cloud-based computation and storage

The conceptualization that the acquisition node is only capable of AES encryption implies that an AES encrypted version of the medical records will be permanently stored in the cloud. However, any portion of this data that needs to be operated on in real-time has to be converted to the FHE encrypted version. While AES to FHE conversion can be done securely by using the methods in [20] which is very compute-intensive, this conversion has to be performed only once.

The AES to FHE conversion can be performed offline, by using the least expensive resources, since the conversion time will be exposed as a delay in providing the remote-monitored patient data to the doctor. This delay might not be important, since the doctor will typically need these results after the monitoring has been completed, e.g., in a few days [22]–[24].

### C. GUI (Thin) end nodes

The other end-node of the application is the GUI device which displays the results to the doctor. The entire set of computations will be executed in the cloud, thus the end result will be in FHE encrypted format when it is transferred to the GUI device. This translates to a necessary capability of the GUI device to perform FHE decryption. As we will demonstrate in Section IV, today's GUI devices are more than capable of performing FHE decryiption efficiently.

## IV. Performance Evaluation

In this section, we will provide actual measured results of calculating the average heart rate of a patient using two FHE schemes: a) Gentry's first FHE scheme [7] and b) the BGV scheme [8]. We use the library in [25] for the former scheme, and the library in [19] for the latter.

### A. Experimental Setup

To calculate the average heart rate of a patient, we use a sample ECG annotation file record from the THEW ECG database [10]. The annotation file is generated from ECG samples of the patient captured via a 12-lead Holter system at a 1000 Hz sampling rate over a 24 hour period and contains 87,896 entries for temporal distance ($toc$) of consecutive heart beats. Each $toc$ value is represented as a 12b number. Based on the information in the annotation file, average heart rate during $N$ heart beats can be calculated in two steps: 1) Accumulating the $toc$ values for $N$ heart beats, 2) dividing the final sum by $N$, followed by a multiplication with sample acquisition time.

The most logical task partitioning is to perform the accumulation in the cloud and leave the final division and multiplication steps to the GUI end-node, based on the observation that, while these final division and multiplications are trivial for the GUI end-node, which can be done after decryption, they are, in fact, computationally expensive in FHE-encrypted format. We ran our simulations on a dual-Xeon E5450 node, 16GB RAM, quad-cores @3GHz.

### B. Results Based on Gentry's Original FHE scheme [7]

In [7] each bit of the message is encrypted individually, generating $m$ ciphertexts for an $m$-bit message. Homomorphic addition and multiplication of ciphertexts will translate into bitwise XOR and AND of the message bits respectively. Table I demonstrates the execution times of each FHE operation.

| Operation | Execution Time |
|---|---|
| Encrypt | 1.45 sec |
| Decrypt | 0.2 sec |
| Recrypt | 24.95 sec |
| Addition | $\leq 1\ \mu$s |
| Multiplication | 1.79 ms |

TABLE I
Execution times for the Gentry's original FHE scheme [7]

To perform additions with the bitwise arithmetic, we choose to implement a Ripple Carry Adder. For each bit, first sum and carry is calculated, then the carry is propagated to the next level. During the carry computation, homomorphic multiplication is performed, which increases the noise in the ciphertext. Therefore, we recrypt the carry before forwarding to the next level to prevent decryption errors. Since recrypt operation takes longer than homomorphic addition and multiplication, 99.9% of the execution time is spent during recryption.

To investigate computational and storage requirements of using the FHE scheme in [7], we choose our case study as calculating average heart rate of the patient during one hour time period. One hour of patient record contains ≈4,096 $toc$ values. To accumulate 4,096 12b numbers, we pick our accumulator size as 24b to prevent an overflow. Based on our simulations on the cluster node, computing a one-hour average heart rate takes ≈700 hours. Each ciphertext encrypts one bit information and has a size of ≈0.1MB. This is equal to an $\approx 800,000\times$ storage expansion ratio and a one hour patient record requires ≈4.8 GB of storage space. Our results indicate that Gentry's original FHE scheme [7] is clearly impractical even for the long-term patient monitoring both in terms of computation and storage.

### C. Results Based on BGV scheme [8]

The BGV scheme [8] and its implementation [19] is one of the most recent and promising works for making FHE more practical. The scheme is based on Ring-LWE [26] where messages and ciphertexts are defined over polynomial rings. Homomorphic addition and multiplication of ciphertexts will correspond to ring additions and multiplications of messages, respectively. Table II shows the execution times for each FHE operation on the cluster node.

| Operation | Execution Time |
|---|---|
| Encrypt | 1.65 sec |
| Decrypt | 0.65 sec |
| Addition | 0.11 ms |
| Multiplication | 0.8 sec |

TABLE II
Execution times for the BGV scheme [8]

The BGV scheme introduces several methods to address inefficiencies of the previous FHE schemes. Multiple messages can be packed into one ciphertext by using the techniques in [27], where each message can carry multi-bit information. The expensive recrypt operation is avoided by defining the level of the function (i.e., multiplicative depth) beforehand and adjusting the parameters during the key generation [8].

To investigate the performance of the BGV scheme, we choose to calculate average heart rate of the patient during a 24-hour time period. For encoding each $toc$ value, we use the methods in [16]. By setting parameters correctly, accumulation can be performed by using only homomorphic additions. We pack each ciphertext with 200 $toc$ values and accumulating the 87,896 $toc$ values can be performed by $\lceil 87{,}896/200 \rceil = 440$ additions. Our simulations show that the 24-hour average heart can be calculated in ≈70 ms. Each ciphertext, containing 200 *toc* values (12b each), occupies ≈65KB storage space which corresponds to a data expansion ratio of $\frac{65{,}000\times 8}{200\times 12} \approx 217\times$. Storing a 24-hour patient record requires ≈28MB. While computing the average is slow compared to its unencrypted version, the BGV scheme is very close to providing the result in real-time with acceptable storage expansion.

To explore the maximum attainable acceleration due to parallelism in the cloud, we performed the following experiment: Since the library in [19] is not thread-safe, we explored parallelism at the process level: we run multiple concurrent processes and let the operating system (OS) distribute each thread to available processor threads. The results from each core/thread can be combined later through OS-level pipes. Table III demonstrates our experimental results: The *Speed-up* and *Efficiency* columns indicate the maximum acceleration attainable due to hardware parallelism.

| Processes | Runtime (ms) | Speed-up | Efficiency (%) |
|---|---|---|---|
| 1 | 69.8 | 1.00 | 100 |
| 2 | 35.8 | 1.95 | 97.4 |
| 4 | 21.5 | 3.25 | 81.4 |
| 8 | 11.75 | 5.96 | 74.5 |

TABLE III
MULTI-PROCESS RUNTIME USING A DUAL-SOCKET XEON SERVER.

## V. CONCLUSIONS AND FUTURE WORK

We have described a cloud-based system using FHE to review ECG data which presents a novel method of accessing, analysing and displaying of private health information (PHI). We chose to use ECG data for our study due to its ubiquitous use in healthcare, however the system we described maybe generalized to many other medical applications. We identified the two major challenges in turning an FHE-based medical application into reality: storage and computation.

We determined the storage-expansion due to FHE-based data to be 217× in our test study. To cope with this, we proposed a tiered cloud storage mechanism which stores the raw medical data in AES-encrypted format, and a second-tier FHE-data cache is used to store the FHE version of this data. We demonstrated that, this tiered storage is sufficient to allow the ECG-based medical application to store sufficient patient information to allow long-term health monitoring.

We determined that, simple operations, such as, computing the average heart rate over a 24-hour period can be done in real time with a 70 ms latency using the newly introduced FHE library based on the Ring LWE algorithm. We also demonstrated the applicability of hardware parallelism to FHE computations. This simple, yet real, operation, combined with the ability of parallelizing using clustered hardware, shows the possibility of turning FHE into reality for a restricted set of medical cloud applications in the near future. Although far from a complete feasibility study, we argue that, our preliminary work sets the stage for tackling significantly more sophisticated medical applications in the near future.

## ACKNOWLEDGMENT

This work was supported in part by the National Science Foundation grant CNS-1239423 and a gift from Nvidia Corp.

## REFERENCES

[1] Suave Lobodzinski and Michael Laks, "New devices for every long-term ecg monitoring," *Cardiology Journal*, vol. 19, no. 2, pp. 210–214, 2012.
[2] Amazon, "Amazon Web Services (AWS)," http://aws.amazon.com.
[3] Microsoft, "Windows Azure," http://www.microsoft.com/windowazure.
[4] Google, "Google App Engine," http://code.google.com/appengine.
[5] Cardio Leaf, "World's Thinnest 3-Lead ECG Patch," http://http://www.clearbridgevitalsigns.com/brochures/CardioLeaf _ULTRA_Brochure.pdf.
[6] National Institute of Standards and Technology, "Advanced encryption standard (AES)," Nov. 2001, FIPS-197.
[7] Craig Gentry, "Fully homomorphic encryption using ideal lattices," 2009, STOC, pp. 169–178.
[8] Zvika Brakerski, Craig Gentry, and Vinod Vaikuntanathan, "(leveled) fully homomorphic encryption without bootstrapping," in *ITCS*, 2012, pp. 309–325.
[9] "PROgramming Computation on EncryptEd Data (PROCEED)," http://www.darpa.mil/Our_Work/I2O/Programs/ PROgramming_Computation_on_EncryptEd_Data_(PROCEED).aspx.
[10] J.P. Courderc, "The telemetric and holter ecg warehouse initiative (thew): A data repository for the design, implementation and validation of ecg-related technologies," *IEEE Publishing*, vol. 6252, no. 5, pp. 6252–6255, 2010.
[11] Marten van Dijk, Craig Gentry, Shai Halevi, and Vinod Vaikuntanathan, "Fully homomorphic encryption over the integers," in *EUROCRYPT*, 2010, pp. 24–43.
[12] Zvika Brakerski and Vinod Vaikuntanathan, "Fully homomorphic encryption from ring-lwe and security for key dependent messages," in *CRYPTO*, 2011, vol. 6841, p. 501.
[13] Zvika Brakerski and Vinod Vaikuntanathan, "Efficient fully homomorphic encryption from (standard) lwe," in *FOCS*, 2011, pp. 97–106.
[14] Jean-Sébastien Coron, Avradip Mandal, David Naccache, and Mehdi Tibouchi, "Fully homomorphic encryption over the integers with shorter public keys," in *CRYPTO*, 2011, pp. 487–504.
[15] Craig Gentry and Shai Halevi, "Fully homomorphic encryption without squashing using depth-3 arithmetic circuits," in *FOCS*, 2011, pp. 107–109.
[16] Michael Naehrig, Kristin Lauter, and Vinod Vaikuntanathan, "Can homomorphic encryption be practical?," in *CCSW*, 2011, pp. 113–124.
[17] Nigel P. Smart and Frederik Vercauteren, "Fully homomorphic encryption with relatively small key and ciphertext sizes," in *PKC*, 2010, pp. 420–443.
[18] Damien Stehlé and Ron Steinfeld, "Faster fully homomorphic encryption," in *ASIACRYPT*, 2010, pp. 377–394.
[19] Shai Halevi and Victor Shoup, ," https://github.com/shaih/HElib.
[20] Craig Gentry, Shai Halevi, and Nigel P. Smart, "Homomorphic evaluation of the AES circuit," in *CRYPTO*, 2012, pp. 850–867.
[21] "Microchip Embedded Security," http://www.microchip.com/pagehandler/en_us/technology/embeddedsecurity.
[22] Tolga Soyata, Rajani Muraleedharan, Colin Funai, Minseok Kwon, and Wendi Heinzelman, "Cloud-Vision: Real-Time face recognition using a Mobile-Cloudlet-Cloud acceleration architecture," in *Proceedings of the 17th IEEE Symposium on Computers and Communications (IEEE ISCC 2012)*, Cappadocia, Turkey, Jul 2012, pp. 59–66.
[23] Tolga Soyata, R. Muraleedharan, S. Ames, J. H. Langdon, C. Funai, M. Kwon, and W. B. Heinzelman, "Combat: mobile cloud-based compute/communications infrastructure for battlefield applications," in *Proceedings of SPIE*, May 2012, vol. 8403, pp. 84030K–84030K.
[24] Tolga Soyata, He Ba, Wendi Heinzelman, Minseok Kwon, and Jiye Shi, "Accelerating mobile cloud computing: A survey," in *Communication Infrastructures for Cloud Computing*, H. T. Mouftah and B. Kantarci, Eds. IGI Global, Hershey, PA, USA, 2013.
[25] Craig Gentry and Shai Halevi, "Implementing gentry's fully-homomorphic encryption scheme," 2011, EUROCRYPT, pp. 129–148.
[26] Vadim Lyubashevsky, Chris Peikert, and Oded Regev, "On ideal lattices and learning with errors over rings," in *EUROCRYPT*, 2010, pp. 1–23.
[27] Nigel P. Smart and Frederik Vercauteren, "Fully homomorphic SIMD operations," Manuscript at http://eprint.iacr.org/2011/133, 2011.