ORIGINAL ARTICLE

Cloud-Based Privacy-Preserving Remote ECG Monitoring and Surveillance

Alex Page, B.S.,* Ovunc Kocabas, M.S.,* Tolga Soyata, Ph.D.,* Mehmet Aktas, M.D.,† and Jean-Philippe Couderc, Ph.D.‡

*From the *Electrical and Computer Engineering Department, University of Rochester, Rochester, NY; †Cardiology Department, University of Rochester Medical Center, Rochester, NY; and ‡Heart Research Follow Up Program, University of Rochester Medical Center, Rochester, NY*

Background: The number of technical solutions for monitoring patients in their daily activities is expected to increase significantly in the near future. Blood pressure, heart rate, temperature, BMI, oxygen saturation, and electrolytes are few of the physiologic factors that will soon be available to patients and their physicians almost continuously. The availability and transfer of this information from the patient to the health provider raises privacy concerns. Moreover, current data encryption approaches expose patient data during processing, therefore restricting their utility in applications requiring data analysis.

Methods: We propose a system that couples health monitoring techniques with analytic methods to permit the extraction of relevant information from patient data without compromising privacy. This proposal is based on the concept of fully homomorphic encryption (FHE). Since this technique is known to be resource-heavy, we develop a proof-of-concept to assess its practicality. Results are presented from our prototype system, which mimics live QT monitoring and detection of drug-induced QT prolongation.

Results: Transferring FHE-encrypted QT and RR samples requires about 2 Mbps of network bandwidth per patient. Comparing FHE-encrypted values—-for example, comparing QTc to a given threshold—runs quickly enough on modest hardware to alert the doctor of important results in real-time.

Conclusions: We demonstrate that FHE could be used to securely transfer and analyze ambulatory health monitoring data. We present a unique concept that could represent a disruptive type of technology with broad applications to multiple monitoring devices. Future work will focus on performance optimizations to accelerate expansion to these other applications.

Ann Noninvasive Electrocardiol 2014;00(0):1–10

telemedicine; cybersecurity; electrocardiogram; e-health

With the miniaturization of health monitoring systems and their integration into everyday devices such as smart phones,[1–3] personal computers, and implantable devices (ICD, ICD-CRT), there is an increasing number of technical solutions to access large amounts of data regarding the health state of patients outside the standard clinical settings. However, the transmission of increasing amounts of health information raises ethical and safety concerns related to patient privacy. Current HIPAA regulations list a set of factors defining protected health information (PHI), but it is uncertain if this list will remain sufficient when cross checking techniques of multiple health information sources may generate opportunities to breach patient privacy.

This article describes a system that uses end-to-end encryption to solve the privacy issues of remote monitoring. This solution allows for data analysis in the cloud, without making patient

Address for correspondence: Alex Page, Electrical and Computer Engineering, University of Rochester, 505 Computer Studies Building, Rochester, NY 14627. Fax: 585-220-2073; E-mail: alex.page@rochester.edu

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

DOI:10.1111/anec.12204

information available to the cloud provider. A proof-of-concept system is presented, using ECG-monitoring of QT prolongation based on continuous Holter recordings from healthy individuals and cardiac patients. Also, a long-term research path is proposed to improve the speed (and available number) of operations that can be performed on encrypted data, which could expand the capabilities and clinical use of such technology.

METHODS

We will demonstrate the feasibility of our concept by simulating clinical cases of patients exposed to drug-induced QT prolongation while being continuously monitored using body surface ECGs. We will simulate patient surveillance for the acquired form of the long QT syndrome (LQTS) while providing data storage and limited processing in the cloud using fully homomorphic encryption (FHE), an emerging technique that allows computation on encrypted data (i.e., computation without observing the actual data). Furthermore, we will evaluate the impact of using FHE on the performance of the system for detecting the presence of QT prolongation using real clinical data.

The technological infrastructure exists to allow remote ECG monitoring via a cloud provider; the data rate required to transfer uncompressed ECG recordings is on the order of 10 KB/s,[4] which is easily accommodated by virtually any Wi-Fi access point or cellular network, and storage of encrypted medical data in the cloud is an accepted practice.[5,6] Using existing components, there are essentially three ways to design a remote monitoring system:

(1) The hospital itself can maintain its own datacenter, but this is costly in terms of hardware, utilities, personnel, and space.[7,8] Locally maintained datacenters also limit scalability and interoperability.[9]
(2) Servers can be rented from a cloud provider that's willing to sign a business associate agreement (BAA).[9–11] However, this restricts the hospital's options. Furthermore, while such an agreement ensures HIPAA compliance, it cannot guarantee privacy; breaches may happen even when best practices are followed.
(3) The cloud can be used in a "storage only" mode, that is, without the ability to analyze data. Data can be encrypted to prevent any third parties from reading it, adding a layer of security to option (2) and also enabling the use of "untrusted" cloud providers. (In the context of this article, "untrusted" providers are providers who should not have the ability to read the data that they are hosting.) This type of system provides only the $\mathcal{A}$–$\mathcal{B}$–$\mathcal{C}$ chain in Figure 1, and is the typical configuration for services such as EVault[5] or CareCloud.[12]

Thus, while it is *technically* possible to design a remote-monitoring system using existing tools, it is typically *not* possible to do so within the guidelines of HIPAA without sacrificing computational capabilities or incurring large additional costs. While we can *store* encrypted PHI in the cloud, we have no method to securely *process* it—for example, to generate reports or alerts—on an untrusted cloud platform. Therefore, our aims are: (1) to define a system that solves the HIPAA/privacy issues with remote patient monitoring and (2) to demonstrate a proof-of-concept for this system using a clinically relevant application.

ECG Monitoring Using Fully Homomorphic Encryption (FHE)

As alluded to above, conventional cryptography has a major limitation: it precludes data analysis in the cloud (unless the decryption key(s) are also available to the cloud provider). That is, most basic arithmetic operations will not yield correct results—or even meaningful results—when applied to data that has been encrypted by a standard algorithm such as the advanced encryption standard (AES).[13] To overcome this severe limitation, we look to FHE.[14] FHE is a novel encryption method that allows analysis (i.e., arithmetic operations) to take place on encrypted data, yielding correct, encrypted results. Using this technique, a server can analyze data and provide results to a doctor, *without the server ever knowing what the data or results were*; the doctor is only able to view the results because (s)he possesses the decryption key. This capability does not come without a cost, though, the set of mathematical operations that can be performed on FHE-encrypted data is still somewhat limited, and calculations can be very

time and memory consuming. Nevertheless, our initial results indicate that FHE is already practical for some basic applications, and that there is great potential for extending and accelerating the available functions in the future.

We propose a system that allows ECG data to be: aggregated on a smartphone (or PC) in the vicinity of a patient, uploaded to a cloud services provider (such as Amazon Web Services[15] or Google Cloud Platform[16]), analyzed in the cloud (with results forwarded to the doctor), and protected along the entire path using a combination of standard encryption techniques and FHE. The doctor can review the results (e.g., an annotated ECG waveform), release the patient's diagnosis, and decide on a course of action.

There are three main parts to the proposed system, shown in Figure 1, which will be described below. Many of these pieces have already been implemented and tested in our lab, and will be described further in the "Implementation" section:

(1) The sensor(s) and embedded system(s) in the vicinity of the patient ($\mathcal{A}$, $\mathcal{B}$, $\mathcal{D}$, and $\mathcal{E}$ in Fig. 1).
(2) The servers and storage located at a large datacenter ("Cloud" in Fig. 1).
(3) The doctor's computer(s) (performing $\mathcal{C}$ and $\mathcal{G}$).

The patient wears sensors ($\mathcal{A}$) that transmit wirelessly and securely (e.g., using AES[17]) to a nearby smartphone. The sensor system may be, for example, an ECG patch with an embedded microprocessor and Bluetooth transceiver. The phone decrypts the incoming sensor data, performs some preliminary analysis ($\mathcal{D}$), re-encrypts the data using two different techniques—one conventional ($\mathcal{B}$), the other using FHE ($\mathcal{E}$)—then uploads it to the cloud. We assume that the hospital provides a dedicated phone for this purpose; this way, factors such as free storage/memory, battery life, security, and data rate/caps are under the hospital's control, and not dependent on the patient's particular hardware and service plan. So, when we refer to the "patient's" phone or PC, we really mean the one the hospital issued.

A server hosts an upload directory (via a secure protocol such as SFTP) for patients' devices to send their data. The host may have signed a BAA, but this is not necessarily a requirement since they will not actually have access to the unencrypted PHI. When new data is received, it will be one of two types: (1) conventionally encrypted raw data or (2) FHE-encrypted, preprocessed data. In case (1), the server simply holds the data until the doctor asks for it. In case (2), the server finishes analysis of the data, and generates results such as "QT prolongation warning" (all in $\mathcal{F}$).

An application on the healthcare provider's tablet, smartphone, or PC will actively decrypt the results from the server (in $\mathcal{G}$), and notify the doctor of any changes in a patient's status for which they had requested an alert. In response to these notifications—or at any other time—the doctor can request more information. For example, the doctor may be notified that QTc exceeded a given threshold for a 30-second interval. The doctor would then request to see the patient's ECG waveform for that 30-second interval. This waveform would be retrieved from the server, and decrypted by $\mathcal{C}$ for display. (The doctor will be able to initialize and configure sensors, encryption keys, alerts, etc. for their patients via a "one-stop" administration website.)

The Cloudlet: Providing Postacquisition Assistance

Because $\mathcal{D}$ and $\mathcal{E}$ (in Fig. 1) may be overwhelming for a smartphone—especially when battery life is considered—it could be desirable to bypass the phone, replacing it with a nearby PC (connected to the patient's home power and Internet) instead. A fast, local computer in this "cloud interface/helper" configuration is sometimes referred to as a "cloudlet."[18–21]

There are many advantages to this approach; most importantly, a cloudlet is more capable than a phone in terms of hardware—both for speed, and extensibility. It is also likely to have access to a faster, more reliable Internet connection, and the patient does not have to carry it or charge it. And cost-wise, a computer such as an Intel NUC[22] would be comparable to the smartphone approach. The primary disadvantage of a cloudlet compared to a phone is that it limits the mobility of the patient, requiring that they stay within range of a fixed location. Another possible disadvantage is setup time; since some patients will not know how to connect a new computer to their home network, a technician may need to set the cloudlet up for them. (This could potentially be solved by using

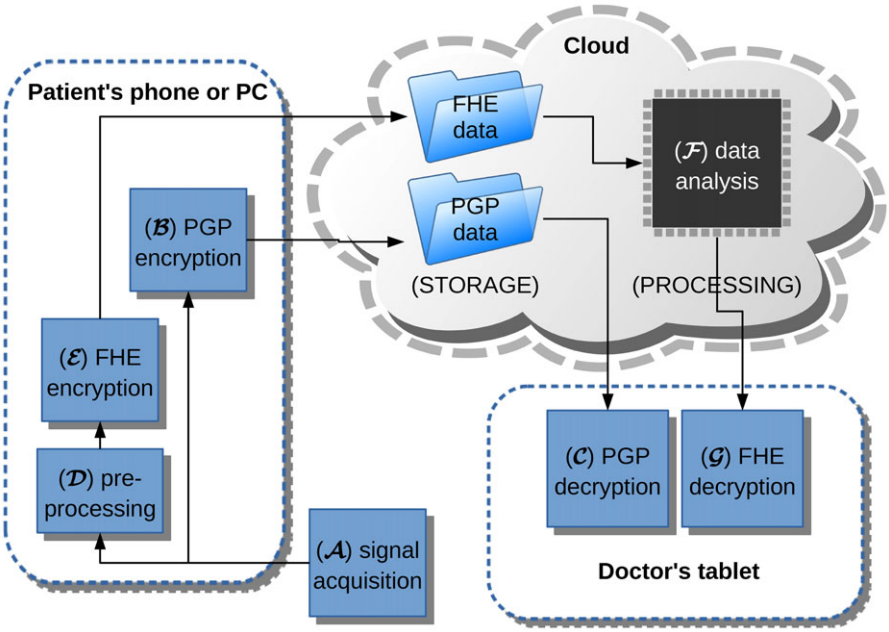

Figure 1. Overview of proposed system to enable privacy-preserving computation. $\mathcal{A}$ is a wireless ECG patch (or some other sensor), and $\mathcal{C}$ allows the doctor to view its output. The preprocessing step $\mathcal{D}$ is in place to circumvent current limitations of HElib (explained in the "Fully Homomorphic Encryption" section), that is, it provides an opportunity to compute values *before* FHE encryption that would be hard to compute *after* encryption. Systems providing $\mathcal{A}$–$\mathcal{B}$–$\mathcal{C}$ already exist; the addition of the $\mathcal{E}$–$\mathcal{F}$–$\mathcal{G}$ chain is the focus of our research. Without this chain, the cloud cannot securely perform any kind of signal analysis.

a 4G cellular card in the cloudlet, rather than the patient's Internet connection.)

It is also possible to use the patient's *own* computer as the cloudlet, that is, to install an application on their home computer that performs $\mathcal{B}$, $\mathcal{D}$, and $\mathcal{E}$, and receives sensor data via, for example, a USB Wi-Fi adapter. However, as there are no guarantees of speed, security, or reliability on a patient's personal computer, it is recommended instead that the hospital maintain a set of preconfigured cloudlet PCs (similar to the smartphone distribution plan mentioned in the "ECG Monitoring Using Fully Homomorphic Encryption" section). These PCs would be issued for remote-monitoring sessions, then wiped/reconfigured for the next patients.

LQTS Surveillance

Prolongation of the QT/QTc interval is an accepted surrogate marker of an increased risk for life-threatening events.[23] As of today, there are hundreds of drugs available on the U.S. market that can slightly prolong the QT interval. While these drugs are generally safe, the accumulation of their small QT effects when patients are prescribed with multiple drugs can become a health concern. The acquired LQTS is modulated by more than just drug interaction; the variability in individuals' response, the diet, the circadian variation of heart regulations are amongst a set of factors that can play a crucial role in the patient response to a drug with potential QT effect. Therefore, a solution is to continuously monitor patients and enable QT surveillance. This concept implies that QT intervals are continuously assessed from an ambulatory ECG signal and corrected for heart rate. There are many different methods that are used for "correcting" QT for heart rate. We opted to implement Fridericia's formula,[24] $QTc = \frac{QT}{\sqrt[3]{RR/\text{sec}}}$.

To measure the QT on a beat-to-beat basis, we selected the algorithm developed by Yuriy Chesnokov.[25] This award-winning open-source algorithm was tested on 548 sample ECG recordings and delivered a 17.30 ms RMS error (i.e., approximately 4% error) in QT interval measurement. The

algorithm also provides fiducials for the PR interval and QRS complex for all ECG leads. We ported this code from Windows to Linux, and modified it to read ISHNE-formatted data files[4] from the THEW database.[26] It was then extended to output QT and RR values for use in our FHE comparator. The resulting program currently runs on an Amazon server, and the source code is much more cross-platform than the original—that is, it is closer to being executable on a smartphone or embedded microprocessor, which will be helpful during later development.

Conventional Security

Certain data—namely, the ECG waveform—should be forwarded unaltered from the patient to the doctor. This data does not need to be encrypted using FHE; instead, we can encrypt it with a conventional algorithm. This encrypted waveform will simply be held on the server until the doctor retrieves and decrypts it. In this section, we describe standard cryptographic/security techniques that will be used to accomplish this. Our requirements are:

(1) Algorithms should be lightweight in terms of storage and processing.
(2) The overall configuration should provide as much security as possible in "worst case" scenarios (e.g., a lost or stolen phone/tablet).
(3) Patient data should be easily transferred between doctors when authorized.
(4) Patients should not be able to see other patients' data.

One system that satisfies these requirements is PGP,[27] which is typically used to secure email communications. PGP compresses data and uses AES for encryption, which is good for requirements 1 and 2. Communications are "wrapped" with RSA[28]—a crypto algorithm that helps secure much of the Internet—also satisfying 2. By providing only *public* keys to the patients, we satisfy both 2 and 4. Finally, since PGP-encrypted data contains a header of "allowed recipients," transferring an encrypted patient file to another doctor is simply a matter of (an approved doctor) adding the new doctor to the file header (satisfying 3).

On the server receiving PHI, file permissions should be set to prevent a patient from downloading or viewing any other files. Also, to ensure that only intended patients are able to access the server, we recommend using the SFTP protocol. A unique key can be assigned to patient for a particular monitoring session, and then revoked/removed from the server. This protocol also adds an extra layer of security to the system, as SFTP not only authenticates clients,[29] but encrypts data transfers[30] as well. (Note that these security steps are somewhat redundant, as the files being transferred are already encrypted. However, they help mitigate attacks that could corrupt data or degrade service.)

Steps must be taken to ensure that PHI is protected at the phone/tablet level, as well; for example, keeping sensor data in RAM rather than on an SD card makes it harder for an attacker to retrieve. Further, phones should be wiped between patients, sensor keys should be rotated, and other details of key distribution must be planned in such a way to minimize privacy risks. And as always, the doctor must protect their password.

Fully Homomorphic Encryption

As we have discussed, conventional encryption is useful when a server is storing data for a doctor to retrieve. It is *not* useful if the (untrusted) server needs to perform any calculations—identifying a patient's heart rate from raw ECG data, for example. These tasks are impossible because basic mathematical operations such as addition or multiplication cannot be applied to conventionally encrypted data in a meaningful way. Since data analysis in the cloud is a key goal of our system, we must choose (or create) an FHE library to be used for this task. While there are a few options, the de facto library for this is *HElib*.[31]

HElib was created in 2013 and is based on the Brakerski–Gentry–Vaikuntanathan (BGV) encryption scheme.[32] It provides the ability to encrypt/decrypt data homomorphically, and includes several mathematical functions (such as addition and multiplication) that can operate on the encrypted data. On top of HElib's default primitives, we have built new functions to allow for computation of statistics (such as the average value for a list[33,34]) and comparison of values.

There are a couple of key concepts that must be kept in mind when working with data in HElib: (1) every algorithm you want to run has a "depth" or "level," which is essentially an indicator of its complexity, and (2) you can "pack" more than

one value into a single variable/ciphertext. These ideas have important implications for us: the depth determines the speed and memory requirements of the algorithm, and the ability to pack many values into a ciphertext grants us some parallel computation capabilities.

The FHE comparison operator (>) is used in our prototype system to compare QTc to a threshold. This operator has some special requirements. One important limitation is that only *unsigned integers* can be compared. Further, since we would like to look at median QTc for about 40–50 heartbeats at a time (in order to filter out noisy/erroneous data), it could be desirable to pack that many heart beats into a single ciphertext for HElib to operate on. The "integer" requirement is important to keep in mind, as converting floating-point inputs to integers will cause the results to be less precise (especially if they are also dropped from, e.g., 32 bits to 16 bits).

Implementation

A prototype system has already been developed in our lab. Because a full, general implementation is so complex, we chose to focus on a single proof-of-concept application: detecting prolonged QT interval in ambulatory condition. The current components of this system are illustrated in Figure 2.

Rather than capturing data from a live ECG sensor, prerecorded ECG data from the University of Rochester's THEW database[26] is used. An application on an Android-based phone uploads this ECG data to an Amazon EC2 server. This communication is authenticated and encrypted by the SSH protocol.

The flow of data in the cloud proceeds as follows:

(1) A Linux-based server running on an Amazon EC2 instance receives the patient's ECG data—an ISHNE-formatted file—over an SFTP connection.
(2) The delineation program (from the "LQTS Surveillance" section) takes the raw ECG file, and annotates QT and RR. (This algorithm is too complex to be performed under FHE at this point, so in the proof-of-concept, the server is running it on the unencrypted data.)
(3) For each heartbeat, the FHE comparison "QTc > threshold?" is performed (where "threshold" is a value such as 0.470 seconds). If FHE is not required (i.e., on a trusted server), the comparison can be performed very quickly and accurately by bypassing HElib. Using HElib, though, the process is much slower, and some precision is lost due to the QT and RR values being scaled and converted to integers—see the "FHE-based Prolonged QT Detection" section for more details.
(4) The results are encrypted using PGP, and emailed to the doctor. At the moment, results are reported for every heartbeat, but in the future we will filter out noise by only looking at the median for groups of beats.

Finally, the doctor uses an email client that supports PGP (such as Thunderbird with Enigmail, or K-9 Mail with APG) to open the message containing the results. Development is underway to allow viewing the ECG plot in a smartphone/tablet application.

RESULTS

Using the system in Figure 2, we calculated: the bandwidth that would be required to transfer raw ECG data and FHE-encrypted values to/from the cloud, the processing time for the QT/RR delineation algorithm, and the processing time to detect prolonged QTc using FHE.

Data Rate Requirements

A 3-lead ECG recording at 200 Hz generates approximately 1 KB of data per second, while a 12-lead recording at 1000 Hz generates approximately 23 KB of data per second. Even without compression, these rates are well-within typical Internet bandwidth limitations for a cellular system or home/commercial Internet provider (which covers all expected configurations at the doctor's or patient's end of the system).

However, we also need to transfer FHE-encrypted data to and from the cloud. $\mathcal{D}$ in Figure 1 takes raw ECG data and selects/computes a smaller set of values that should be FHE-encrypted. In our case, these are (equivalent to) QT and RR for each heartbeat. (On the other end of the system—$\mathcal{F}$ and $\mathcal{G}$ in the figure—FHE-encrypted results must be transferred to the doctor. In our application, though, these results require less throughput than the inputs, and therefore do not set any new restrictions on bandwidth.) Assuming an average

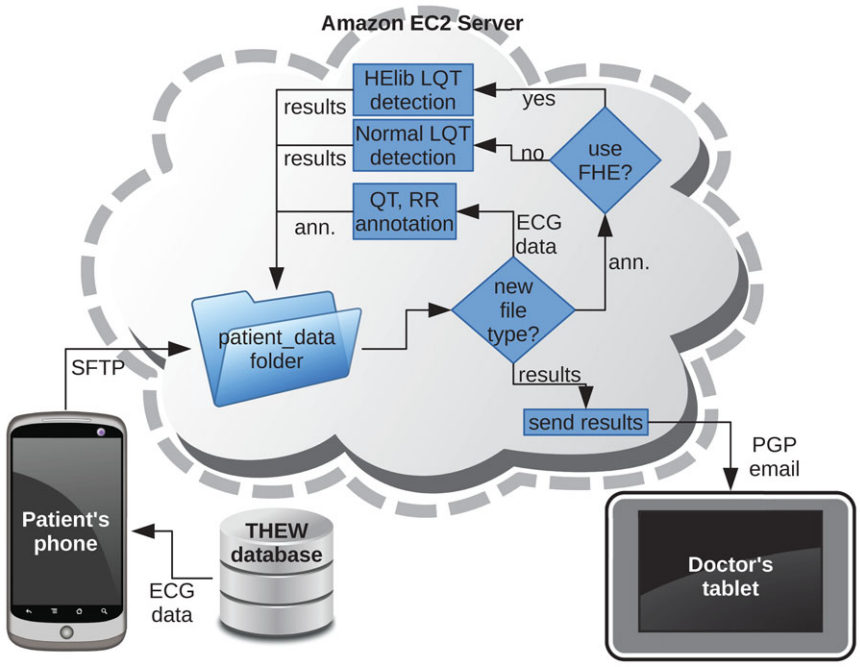

Figure 2. Our current system. Communication links are secure, but several of the functions that need to take place at the endpoints are currently running on the server.

heart rate of 80 bpm, the patient must transmit 2–4 FHE ciphertexts per minute to the server for both QT and RR, since each ciphertext contains QT or RR values for about 20–40 heartbeats. (The exact number of values per ciphertext depends on hardware limitations and HElib parameters. For this particular operation, we are setting the HElib parameter L to values between 10 and 15.) Each ciphertext is 2–3 MB, so the bandwidth requirement for the FHE-encrypted QT and RR values averages about 260 KB/s (i.e., 2 Mbps). This is not an *extremely* prohibitive data rate, but it does require a higher end consumer Internet plan; a 3G cellular data plan, for example, would *not* be able to support it. This level of sustained throughput will also exacerbate the battery life issues we were already anticipating, which suggests that a cloudlet-based approach (described in "The Cloudlet: Providing Postacquisition Assistance") may be best for our next system revision.

QT, RR Delineation

Twenty-four-hour ECG recordings for 202 patients were processed using the program from the "LQTS Surveillance" section. The sample rate for the recordings was 200 Hz, and only the first lead was analyzed. The total processing time on an i7 CPU was approximately 2.2 hours. Based on this timing, delineation of three leads is expected to run about 730× faster than real-time on a desktop or server CPU. On a smartphone, performance will be worse; based on a few standard benchmarks,[35] a typical smartphone CPU is roughly 6–8 times slower than an Intel Core i7. If we conservatively assume, then, that a smartphone will take 8× longer to run the delineation program than a desktop PC, it can still accomplish the task almost 100× faster than real time.

FHE-based Prolonged QT Detection

For the HElib parameters we typically use, with 32-bit precision for QT and RR, running on an i7 processor, FHE comparison can process about 83 heartbeats per minute. At 16 bits, we lower the data rate and speed up the processing to roughly 220 heartbeats per minute, but lose precision. Either way, this operation is fast enough to keep up with the data streaming from the patient.

Because of the clear advantage of 16-bit precision in terms of data rate and CPU speed, we prefer

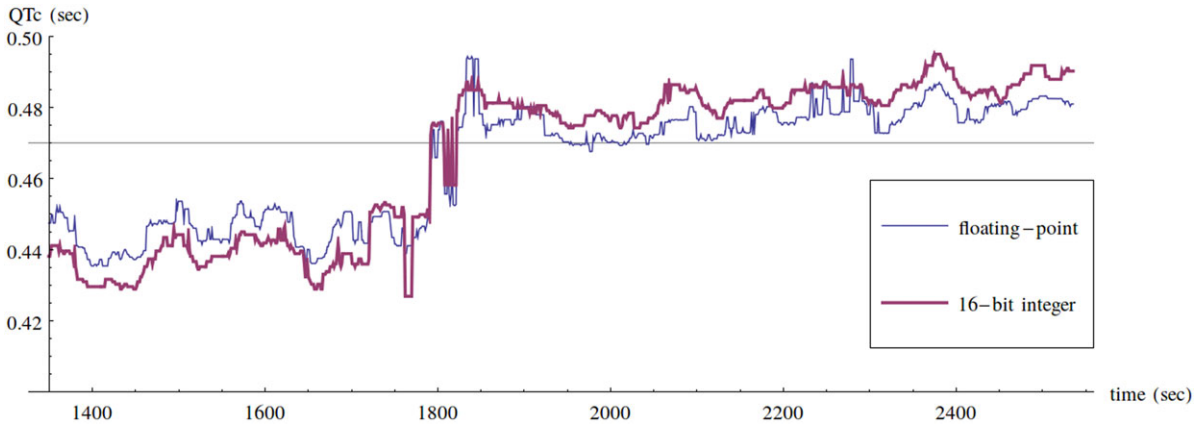

Figure 3. Impact of converting 32-bit floating-point QT and RR values to 16-bit integers prior to FHE-based QTc evaluation. Both lines were median-filtered with a radius of 10 heart beats. (The line for 32-bit integers is not shown, as it is virtually indistinguishable from the floating-point version.)

it over 32-bit. However, we must quantify the precision loss (mentioned in the "Fully Homomorphic Encryption" section) at 16 bits to verify that we still receive correct, useful results. For 32-bit QT and RR values, the `float` → `int` conversion process introduces only 0.25 ms RMS error to QTc. (Error was computed as the RMS of the differences between corresponding integer and floating-point QTc values.) When using 16-bit messages, though—that is, `short int` rather than `int`—the error increased to about ±10 ms RMS. These results were computed across >17 million heartbeats. Figure 3 illustrates this error in computed QTc values during continuous monitoring of a female subject exposed to a IV dose of sotalol (2 mg/kg body weight) at approximately 30 minutes into the recording (see database E-OTH-12–0006–009 from the THEW). This example demonstrates the ability of our method to monitor QT and to detect QT prolongation (above threshold line) that would trigger an alarm message to the subject's physician.

DISCUSSION AND FUTURE WORK

Homomorphic encryption techniques preserve patient privacy in the cloud, at the cost of additional computational resources. We have shown that while these resource costs can be quite daunting, FHE is already practical for a basic application. In our proof-of-concept, we detect prolonged QTc from QT and RR samples in real-time. Several tradeoffs are present, including precision and privacy versus computational efficiency and speed.

Much work remains to improve the realism and practicality of our prototype. Most importantly, we must port HElib and other libraries to run on smartphones/tablets rather than in the cloud. The main reason that this has not been completed yet is simply that the source code and prerequisite libraries for our chosen QT/RR delineation and homomorphic encryption algorithms are written in C++; this code is therefore not (trivially) compatible with Android or iOS. The next stage of development will focus on porting these libraries to Android, either by (a) rewriting them in Java, (b) taking advantage of the Java JNI to load C++ libraries onto Android, (c) loading an alternate operating system such as Debian onto a smartphone, (d) using a cloudlet rather than a phone, which would support all required libraries natively, or (e) some combination of these. Once all of our libraries are running on a phone or cloudlet, the cloud will be operating in a fully homomorphic mode. From this point, we will be adding primitives to our FHE library and slowly begin transferring more work back to the cloud. The primary focus of our research at this stage will be acceleration techniques (e.g., via parallel programming, cache optimization, GPUs, and/or FPGAs) to make the increasing number of FHE operations practical in the cloud. These performance improvements will allow for expansion into other applications in both medical and non-medical fields. In addition to the FHE-oriented development path, we must also build a suite of applications giving the doctor (and hospital) control of the remote-monitoring process. Finally,

for system completeness, we intend to acquire *live* ECG signals, as opposed to using prerecorded signals from the THEW database.

It is noteworthy that the delivery of large amounts of monitoring data to health care providers represents a challenge; a cardiologist would probably *not* need to receive a daily ECG from all his patients, yet this cardiologist *would* be interested in learning about changes in his patients' cardiac status. Therefore, the development of monitoring systems must be coupled with intelligent algorithms appropriately warning physicians. Our concept proposes to manage health information securely while delivering the information that health providers need to be aware of. We have described an example of a QT interval warning system, but one could easily apply such concept to any other physiological markers of risk.

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

REFERENCES

1. AliveCor Heart Monitor [Internet]. 2014; Available at: http://www.alivecor.com/home. Last accessed October 15, 2014.
2. eMotion ECG [Internet]. 2014; Available at: http://www.megaemg.com/products/emotion-ecg/. Last accessed October 15, 2014.
3. Clearbridge VitalSigns CardioLeaf PRO [Internet]. 2013; Available at: http://www.clearbridgevitalsigns.com/pro.html. Last accessed October 15, 2014.
4. Badilini F. The ISHNE holter standard output file format. Ann Noninvas Electrocardiol [Internet] 2006;3(3):263–266. Available at: http://thew-project.org/papers/Badilini.ISHNE.Holter.Standard.pdf.
5. EVault [Internet]. 2014; Available at: http://www.evault.com/. Last accessed October 15, 2014.
6. 45 CFR 164.304 [Internet]. U.S. Superintendent of Documents, Washington, DC, 20402–0001: Office of the Federal Register, National Archives and Records Administration, 2007. Available at: http://www.gpo.gov/fdsys/pkg/CFR-2007-title45-vol1/pdf/CFR-2007-title45-vol1.pdf. Last accessed October 15, 2014.
7. Patel CD, Shah AJ. Cost model for planning, development and operation of a Data Center [Internet]. HP Laboratories Palo Alto, 2005. Available at: http://www.hpl.hp.com/techreports/2005/HPL-2005-107R1.pdf. Last accessed October 15, 2014.
8. Reichman A. File Storage Costs Less In The Cloud Than In-House. Forrester Research, Cambridge, MA, 2011.
9. Munro D. HIPAA Support Widens In Cloud Vendor Community. Forbes [Internet] 2013; Available at: http://www.forbes.com/sites/danmunro/2013/05/01/hipaa-support-widens-in-cloud-vendor-community/. Last accessed October 15, 2014.
10. Amazon Web Services Compliance [Internet]. 2014; Available at: https://aws.amazon.com/compliance/#hipaa. Last accessed October 15, 2014.
11. HIPAA Compliance with Google Apps [Internet]. 2014; Available at: https://support.google.com/a/answer/3407054?hl=en&ctx=go. Last accessed October 15, 2014.
12. CareCloud [Internet]. 2014; Available at: http://www.carecloud.com/hipaa-compliant-cloud-storage/. Last accessed October 15, 2014.
13. Homomorphic Encryption Breakthrough [Internet]. 2009; Available at: https://www.schneier.com/blog/archives/2009/07/homomorphic_enc.html. Last accessed October 15, 2014.
14. Rivest R, Adleman L, Dertouzos M. *On Data Banks and Privacy Homomorphisms*. Foundations of Secure Computation [Internet], 1978; Available at: http://people.csail.mit.edu/rivest/RivestAdlemanDertouzos-OnDataBanksAndPrivacyHomomorphisms.pdf.
15. Amazon Web Services [Internet]. 2014; Available at: http://aws.amazon.com/. Last accessed October 15, 2014.
16. Google Cloud Platform [Internet]. 2014; Available at: https://cloud.google.com/. Last accessed October 15, 2014.
17. Advanced Encryption Standard (AES) [Internet]. United States National Institute of Standards and Technology, Gaithersburg, MD: 2001. Available at: http://csrc.nist.gov/publications/fips/fips197/fips-197.pdf. Last accessed October 15, 2014.
18. Soyata T, Ba H, Heinzelman W, et al. Accelerating mobile cloud computing: A survey. In Mouftah HT, Kantarci B (eds.): Communication Infrastructures for Cloud Computing. Hershey, PA: IGI Global, 2013, pp. 175–197.
19. Soyata T, Muraleedharan R, Ames S, et al. COMBAT: mobile Cloud-based cOmpute/coMmunications infrastructure for BATtlefield applications. In Proceedings of SPIE. Baltimore, MD: 2012, pp. 84030K–84030K.
20. Soyata T, Muraleedharan R, Funai C, et al. Cloudvision: 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: 2012, pp. 59–66.
21. Wang H, Liu W, Soyata T. Accessing big data in the cloud using mobile devices. In Chelliah PR, Deka G (eds.): Handbook of Research on Cloud Infrastructures for Big Data Analytics. Hershey, PA, USA: IGI Global, 2014, pp. 444–470.
22. Mini PC – Intel NUC [Internet]. 2014; Available at: http://www.intel.com/content/www/us/en/nuc/overview.html. Last accessed October 15, 2014.
23. Fanoe S, Kristensen D, Fink-Jensen A, et al. Risk of arrhythmia induced by psychotropic medications: a proposal for clinical management. Euro Heart J [Internet] 2014; 35(20):1306–1315. Available at: http://eurheartj.oxfordjournals.org/content/early/2014/03/17/eurheartj.ehu100.full. Last accessed October 15, 2014.
24. Fridericia LS. Die Systolendauer im Elektrokardiogramm bei normalen Menschen und bei Herzkranken. Acta Medica Scandinavica 1920;53:469–486.
25. QT Interval Measurement: A challenge from PhysioNet and Computers in Cardiology 2006 [Internet]. 2006; Available at: http://physionet.org/challenge/2006/. Last accessed October 15, 2014.
26. Couderc J. The telemetric and holter ECG warehouse initiative (THEW): a data repository for the design, implementation and validation of ECG-related technologies [Internet]. In Engineering in Medicine and Biology Society (EMBC), 2010 Annual International Conference of the IEEE. IEEE, 2010, pp. 6252–6255. Available at: http://www.thew-project.org/.
27. Callas J, Donnerhacke L, Finney H, et al. OpenPGP Message Format [Internet]. IETF Network Working Group,

2007. Available at: https://tools.ietf.org/html/rfc4880. Last accessed October 15, 2014.
28. Rivest RL, Shamir A, Adleman L. A method for obtaining digital signatures and public-key cryptosystems. ACM [Internet] 1978;21(2):120–126. Available at: http://people.csail.mit.edu/rivest/Rsapaper.pdf.
29. Ylonen T, Lonvick C. The Secure Shell (SSH) Authentication Protocol [Internet]. IETF Network Working Group, 2006. Available at: https://tools.ietf.org/html/rfc4252. Last accessed October 15, 2014.
30. Bellare M, Kohno T, Namprempre C. The Secure Shell (SSH) Transport Layer Encryption Modes [Internet]. IETF Network Working Group, 2006. Available at: https://tools.ietf.org/html/rfc4344. Last accessed October 15, 2014.
31. HElib [Internet]. 2014; Available at: https://github.com/shaih/HElib. Last accessed October 15, 2014.
32. Brakerski Z, Gentry C, Vaikuntanathan V. Fully Homomorphic Encryption without Bootstrapping. 2011.
33. Kocabas O, Soyata T. Medical data analytics in the cloud using homomorphic encryption. In Chelliah PR, Deka G (eds.): Handbook of Research on Cloud Infrastructures for Big Data Analytics. Hershey, PA, USA: IGI Global, 2014, pp. 471–488.
34. Kocabas O, Soyata T, Couderc J-P, et al. Assessment of cloud-based health monitoring using homomorphic encryption. In Proceedings of the 31st IEEE International Conference on Computer Design. Ashville, VA, USA: 2013, pp. 443–446.
35. Blem E, Menon J, Sankaralingam K. A Detailed Analysis of Contemporary ARM and x86 Architectures [Internet]. 2013; Available at: http://research.cs.wisc.edu/vertical/papers/2013/isa-power-struggles-tr.pdf. Last accessed October 15, 2014.