Quantum Homomorphic Encryption

By

Joan Watiri Ngure (njoan@aims.ac.rw)
African Institute for Mathematical Sciences (AIMS), Rwanda

Supervised by: Professor Barry C Sanders

University of Calgary, Canada

June 2018

AN ESSAY PRESENTED TO AIMS RWANDA IN PARTIAL FULFILMENT OF THE REQUIREMENTS FOR THE AWARD OF MASTER OF SCIENCE IN MATHEMATICAL SCIENCES

Contents

1. Classical and Cloud Computing

In this chapter we will discuss about classical computing and its limitations, cloud computing advantages and disadvantages.

1.1 Classical Computing

All the computers we use today are classical. This means that information is stored in bits of 0s and 1s. Operations performed by the computers are faster than actually performing it by hand. For example performing calculations using a calculator as opposed to calculation by hand. The probability of making errors is very high.

A computer is a universal machine. A universal machine can be describe as an electronic device with the ability to solve problems with a solution that can be represented by a set of instructions. These problems are known as computational problems since the classical computer can solve them. Types of computational problems include:

1. Decision Problems - This is similar to making a choice between true or false where true can be represented by 1 and false 0.
2. Search Problems - like looking for a word in a text.
3. Function Problems for instance evaluating $\cos\theta$
4. Optimization Problems - You want the output to be a maximum like maximum profit that can be made, thus you find the inputs that satisfies this condition.

To solve these problems, a computer requires resources. The more resources require the more complex it is described. Thus computational complexity is defined by the resources used.

Computers can be connected to each other in order to communicate.This is known as distributed computing. Each processor has its own memory but can communicate with others by sharing messages. One memory can also be shared by multiple processors (parallel computing). Parallelism speeds up computations since different tasks can be performed at the same time. The more the number of processors the more expensive a machine is.

1.1.1 Advantages of Classical Computing.

1. Tasks are performed faster than performing them by hand like calculations. This leads to an increased productivity. Also your work is more organized and the probability of making errors such as wrong spellings and miscalculations is greatly reduced.
2. Communication has been made easier, and the world is more connected. People can easily interact and share ideas or do other activities together. As you connect with more people your skills are improved.

1.1.2 Limitations of Classical Computing.

1. Even with a computer with multiple processors, tasks are performed sequentially. A processor has to finish one task in order to perform another.
2. In case of theft or crashing of the computer, data is lost. This could lead to great loses.
3. Related tasks cannot be performed at the same time by a single processor. This also makes the performance of the tasks slow since you have to wait for one task to complete then perform the next.
4. When working on a computer that is not connected to other computers, for example personal computers, implies that you have to have your computer whenever you need to perform tasks.
5. It is a requirement for some softwares to keep on upgrading to the latest software versions since there are added functionalities. This may require obtaining licenses. This can be expensive to individuals and companies,especially when you need also to train people on the usage of the new software functionalities.
6. Determined people (mind hackers) may learn your password without your knowledge or even you suspecting anything. You may also forget to sign out from your computer and someone gains access to it. Data is therefore duplicated or stolen hence no security in such instances.

1.2 Cloud Computing

1.2.1 What is Cloud Computing?. Cloud Computing is the process whereby, computation is done somewhere else (in the cloud) and is similar to working on your personal computers (Hayes, 2008). Cloud is some server stored somewhere anonymous only the service providers know about its whereabouts. Clients for example companies or individuals are able to request services such as storage, manipulation of data and computations among others. Cloud services are offered as demanded by the clients. Clients do not have to train their staff, buy new infrastructure or get licenses for softwares.

The three major types of cloud computing services are:

1. Platform as a Service (PaaS) such as Google App Engine. This is mainly used by developers.
2. Software as a Service (SaaS) for example email and Facebook.
3. Infrastructure as a Service (IaaS) like storage facilities and servers.

Examples of companies offering cloud services are Amazon, Google and IBM.

1.2.2 Advantages of Cloud Computing.

1. Provided you are connected to the internet, you can access your data and carryout your computations from any computer irrespective of your location (Miller, 2008) and you can detect any tampering in real time (Subashini and Kavitha, 2011).
2. When using your personal computer, you can only access files stored locally alone, with cloud computing different people can access the same files, hence teamwork and collaboration is made easy (Miller, 2008).
3. As a client/ user, you don't have to worry about incurring costs of updating and upgrading software. The service provider ensures that everything is upto date.
4. Our personal computers are limited to a certain storage capacity and computing power, cloud computing ensures these resources are limitless. Also the tasks are carried out faster.
5. In case your computer crashes, you do not lose the data since it is stored in the cloud. If you decide to change the operating system for example linux to windows, incompatibility issues do not exist.

1.2.3 Disadvantages of Cloud Computing. The two main disadvantages of cloud computing are:

1. Requires connection to the internet always.
2. Security.

We will dwell more on the security issue because this is what we intend to solve with Quantum Homomorphic Encryption.

1.2.4 Security as an issue. This argument is based on (Subashini and Kavitha, 2011) and (Miller, 2008). Security is one of the issues that is making some companies to shy away from cloud computing. For example most governments have rules that their sensitive information should not leave the country. In cloud computing data maybe:

1. Changed. This interferes with its integrity. Integrity can be defined as the quality of wholeness. Data which has been changed is not whole. Data could be changed to manipulate intended results.
2. Duplicated.This amounts to information theft. In cloud computing making an exact copy of data is possible.
3. Interfered with. Since many clients are using the same server, the data may mix up or some clients could be malicious and interfere with other client's data.
4. Leaked. Sensitive information maybe exposed to unintended audience. For example, if a user receives emails from different people but always performs a search from a specific sender, this could reveal to the server that this person interests the client for some reason.

The bottom line is, we do not trust the server. Our privacy, confidentiality and integrity is put at risk. The data may end up falling in the wrong hands with unauthorized personnel which would lead to severe consequences. Organizations whose data is mainly sensitive information, would find it hard to adopt cloud computing. A user can encrypt data before storing it in the cloud and when he/she needs it they can download and decrypt it. Thus this user cannot use other services apart from the storage. Other costs of hardware and software, which will be used to process the data purchasing are incurred(Yang et al., 2013).

2. Quantum Computing and Quantum Homomorphic Encryption

In this chapter we will discuss what quantum computing and Quantum Homomorphic Encryption entails. Also why quantum and not classical.

2.1 Quantum Computing

We have quantum bits(qubits) represented as $|0\rangle$ and $1\rangle$ as opposed to the classical bits in classical computing. In Quantum Computing we have gates namely Hadamard (H), T and CNOT. The gates can be represented as:

$$\text{H} = \tfrac{1}{\sqrt{2}}\begin{pmatrix}1 & 1\\ 1 & -1\end{pmatrix}$$

$$\text{CNOT} = \begin{pmatrix}1 & 0 & 0 & 0\\ 0 & 1 & 0 & 0\\ 0 & 0 & 0 & 1\\ 0 & 0 & 1 & 0\end{pmatrix}$$

$$\text{T}= \begin{pmatrix}1 & 0\\ 0 & e^{\frac{i\pi}{4}}\end{pmatrix}$$

A combination of these gates produces other gates for example: $T^4 =$ Z gate

$$\text{Z}= \begin{pmatrix}1 & 0\\ 0 & -1\end{pmatrix}$$

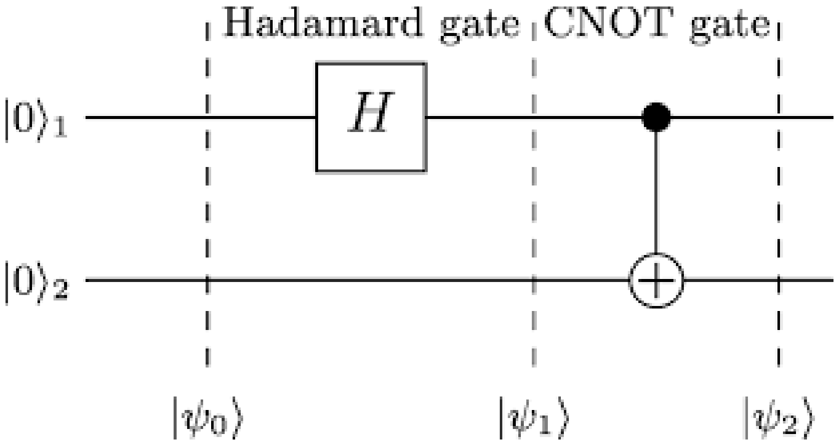
The X gate which is a NOT gate is a combination of HZH.

$$\text{X}= \begin{pmatrix}0 & 1\\ 1 & 0\end{pmatrix}$$

The two main ideas brought about by quantum computing are superposition and entanglement which are adopted from quantum mechanics. Hence if any computation uses these two mechanisms it is quantum computing. In quantum computing, a zero and a one can be represented simultaneously. This is known as superposition. It saves on processing time. For example you have a box full of white chalks but you have one green chalk that you want to obtain from the box. The box is not transparent hence you cannot see what is inside. To solve the problem classically, you have to pull one chalk at a time until you get the green one. Quantum permits you to hold all the chalks at the same time and use probability to pull out the green chalk. This is the idea about superposition.

Entanglement is when two particles interact and acquire states of each other, thus the output will be a mixture of the interacting particles, hence to describe one particle you have to refer to the other. To illustrate an entanglement let's use a Hadamard and CNOT gates. A CNOT gate is a Controlled NOT gate. If the control is qubit of one then we apply a NOT to the bit in the target, otherwise nothing happens. Below is the circuit (fig 1.1):

Figure 2.1: An entangled circuit



At $|\psi_0\rangle$ we have the output as $|0\rangle_1|0\rangle_2$ which can be written as $|0_1 0_2\rangle$. We have $|0\rangle_1$ qubit passing through the Hadamard gate (H). $H|0\rangle = |\bar{0}\rangle = \frac{|0\rangle + |1\rangle}{\sqrt{2}}$. $|0\rangle_2$ does not change. The output at $|\psi_1\rangle$ is $(\frac{|0\rangle + |1\rangle}{\sqrt{2}})|0\rangle = \frac{|0\rangle_1|0\rangle_2 + |1\rangle_1|0\rangle_2}{\sqrt{2}}$. We then the pass through the CNOT. Since $|0\rangle_1$ is 0, $|0\rangle_2$ remains unchanged, $|1\rangle_1$ is 1 hence $|0\rangle_2$ flips to $|1\rangle_2$. Thus the output at $|\psi_2\rangle$ is $\frac{|0\rangle_1|0\rangle_2 + |1\rangle_1|1\rangle_2}{\sqrt{2}} = \frac{|0_1 0_2\rangle + |1_1 1_2\rangle}{\sqrt{2}}$.

This is a maximally entangled state.

Quantum computing also supports parallelism. As opposed to classical computing where different processors required, quantum computing parallelism is done on the same chip. As long as the tasks are related, they are carried out at the same time.For example:

Compute 5^{13}

$$5^{13} = ((5^2)^2)^2 \cdot (5^2)^2 \cdot 5$$

13 in binary can be represented as 1101.

$$13 = 2^0 \cdot 1 + 2^1 \cdot 0 + 2^2 \cdot 1 + 2^3 \cdot 1$$

$$= 5^{1+4+8} = 5 \cdot 5^4 \cdot 5^8$$

In quantum computing all these computations are performed at the same time. A classical computer will first execute 5^2 then $(5^2)^2$ and so on.

2.2 Encryption

The properties of quantum computing enable computations to be performed on encrypted data (hidden). This is the ability for a server to process data delegated to it, without accessing it. The two types of techniques used are:

1. Blind Quantum Computing - Blind actually means you cannot see. This encryption uses the same application. The server is blinded from input, output and computation. This is a technique where server performs computations they do not know on encrypted data and produces results which they do not also know, that is encrypted. Sounds hilarious right?

2. Quantum Homomorphic Encryption - This is a technique where the server performs computations they actually know of on encrypted data, that is, the input and output is encrypted but the computation is known.

2.2.1 Differences between Blind Quantum Computing and Quantum Homomorphic Encryption. (Broadbent et al., 2009), (Fitzsimons, 2017),Ouyang et al. (2015),(Fitzsimons, 2012)

Table 2.1: Differences between Blind Quantum Computing and Quantum Homomorphic Encryption

	Blind Quantum Computing	Quantum Homomorphic Encryption
1	Everything is encrypted (input, output and computation)	Only the input and output are encrypted.
2	Uses information theoretical security protocol (can not be broken in)	Uses computational security protocol (it is hard to break in).
3	It is possible to expand the system based on the client's needs	The system is fixed
4	An interference with the protocol is usually detected because measures are put into place	No measures are put into place hence there is a probability that an interference can go undetected.

2.2.2 Why Choose Quantum Homomorphic Encryption Over Blind Quantum Computing. Information theoretical security used in Blind Quantum Computing is very hard to implement on a classical client hence its implementation is very expensive. (Ouyang et al., 2015). This implies that very few clients will afford it. In most Blind Quantum Computing protocols, the client has some quantum computation requirements, though protocols that do not have this requirement exist (Broadbent et al., 2009).

2.3 Quantum Homomorphic Encryption

As seen earlier, it is possible to delegate tasks (data processing) to a server while the access to the data is not given away. The ability of a server to perform computations on encrypted data solves our security issue.

2.3.1 How the security issue is solved in Quantum Homomorphic Encryption.

1. Since the server does not know the input or output it is impossible to change data to manipulate results, also the integrity of the information is upheld.
2. Data cannot be duplicated because, in quantum, duplication involves measurement and this changes the state thus you will never get the exact copy.
3. Quantum Homomorphic Encryption as we saw earlier uses computational security. It is very hard to break or attack the encrypted data. To break in you require a lot of time. This makes it almost impossible for sensitive information to be leaked.

2.3.2 How Quantum Homomorphic Encryption works. There are four stages involved namely (Broadbent and Jeffery, 2015)

1. Key generation
2. Encryption
3. Evaluation
4. Decryption

All these stages are performed by the classical client.

2.3.3 Key generation and Encryption. Key generation is taking any input maybe a number or an alphabet and generating a pair of public and private key. When the client generates the key, it is used to encrypt information. Encrypted data can be referred to as a cipher text while unencrypted data is called plain text. A plain text is transformed into a cipher text using the public key.

The client can generate as many key pairs as they want to. This means that different public keys can encrypt the data producing different forms of encrypted data. The client is in a position to choose randomly which encryption to send. Since in homomorphic encryption the computation is known to the server, if you always send a certain encryption to perform the same computation, it would be easy for the server to start guessing. Also if the server knows that different encryptions represent the same data, then it would be easy also to guess.

For example, you delegate the server to perform factorization. Factorization problems are easy to verify when given the results but hard to solve. It is very difficult for the server to know exactly which number you are factorizing since the input and output are encrypted. The computation does not reveal sufficient information about the input and output which would make an attack easy apart from the input is a number and the outputs are prime numbers.

How many numbers and prime numbers exist? Will the serve start guessing one by one? How long will it take for the server to actually guess the correct number? It is possible for the server to guess the correct number but the probability is very low maybe 1 out of a million trials (Yang et al., 2013).

2.3.4 Evaluation and Decryption. After key generation and encryption, we evaluate. This is checking the correctness of the algorithm. Use the public key and the cipher text to produce another cipher text. For example let us encrypt g in mod a prime number p. Then we encrypt it again in mod 2. Our cipher text reads $\text{c} = (g \mod p) \mod 2$. We first decrypt c in mod 2 to using the public key to check whether will get $g \mod p$. If we get the desired result then the algorithm is correct. However the time taken to evaluate and the overall cost should be less than the individual cost of key generation, encryption and decryption.

We have our cipher text and private key, we combine them to produce the plain text. This is our decryption process. The decryption is done by the client after they receive the results of the delegated task (Ouyang et al., 2015).

References

Stefanie Barz, Elham Kashefi, Anne Broadbent, Joseph F Fitzsimons, Anton Zeilinger, and Philip Walther. Demonstration of blind quantum computing. *Science*, 335(6066):303–308, 2012.

Michael J Bremner, Richard Jozsa, and Dan J Shepherd. Classical simulation of commuting quantum computations implies collapse of the polynomial hierarchy. In *Proceedings of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, page rspa20100301. The Royal Society, 2010.

Anne Broadbent and Stacey Jeffery. Quantum homomorphic encryption for circuits of low t-gate complexity. In *Annual Cryptology Conference*, pages 609–629. Springer, 2015.

Anne Broadbent, Joseph Fitzsimons, and Elham Kashefi. Universal blind quantum computation. In *Foundations of Computer Science, 2009. FOCS'09. 50th Annual IEEE Symposium on*, pages 517–526. IEEE, 2009.

Joe Fitzsimons. Blind quantum computing and fully homomorphic encryption, 2012. Stackexchange.

Joseph Fitzsimons, Li Xiao, Simon C Benjamin, and Jonathan A Jones. Quantum information processing with delocalized qubits under global control. *Physical review letters*, 99(3):030501, 2007.

Joseph F Fitzsimons. Private quantum computation: an introduction to blind quantum computing and related protocols. *npj Quantum Information*, 3(1):23, 2017.

Craig Gentry. *A fully homomorphic encryption scheme*. Stanford University, 2009.

Brian Hayes. Cloud computing. *Communications of the ACM*, 51(7):9–11, 2008.

He-Liang Huang, Qi Zhao, Xiongfeng Ma, Chang Liu, Zu-En Su, Xi-Lin Wang, Li Li, Nai-Le Liu, Barry C Sanders, Chao-Yang Lu, et al. Experimental blind quantum computing for a classical client. *Physical review letters*, 119(5):050503, 2017.

Cescily Nicole Metzgar. *RSA Cryptosystem: An Analysis and Python Simulator*. PhD thesis, Appalachian State University, 2017.

Michael Miller. *Cloud computing: Web-based applications that change the way you work and collaborate online*. Que publishing, 2008.

Yingkai Ouyang, Si-Hui Tan, and Joseph Fitzsimons. Quantum homomorphic encryption from quantum codes. *arXiv preprint arXiv:1508.00938*, 2015.

Andrew Steane. Quantum computing. *Reports on Progress in Physics*, 61(2):117, 1998.

Craig Stuntz. What is homomorphic encryption, and why should i care?, 2010. Blog.

Subashini Subashini and Veeraruna Kavitha. A survey on security issues in service delivery models of cloud computing. *Journal of network and computer applications*, 34(1):1–11, 2011.

Si-Hui Tan, Joshua A Kettlewell, Yingkai Ouyang, Lin Chen, and Joseph F Fitzsimons. A quantum approach to homomorphic encryption. *Scientific reports*, 6:33467, 2016.

Max Tillmann, Si-Hui Tan, Sarah E Stoeckl, Barry C Sanders, Hubert de Guise, René Heilmann, Stefan Nolte, Alexander Szameit, and Philip Walther. Generalized multiphoton quantum interference. *Physical Review X*, 5(4):041015, 2015.

Yang Yang et al. *Evaluation of somewhat homomorphic encryption schemes*. PhD thesis, Massachusetts Institute of Technology, 2013.