Cryptography Report

On

Proposing a Secure System for usage of Electronic Health Records and Homomorphic Encryption

Submitted To: Dr Ashutosh Bhatia

As a partial fulfilment for the cryptography Course BITS F453 for 1st Semester 2019-2020, BITS Pilani, Pilani Campus.

Submitted By:

Sanjeev Singla

2017A7PS0152P

CONTENTS:

Topic **Page Number**

Project Proposal (As submitted before for reference):

Cryptography Project Proposal

Objective: To propose a solution to the below problem Statement.

Problem Statement: There are 2 mobile applications - Doctor Application and Patient App. The users use the apps to maintain their electronic health records. Patients organise all their documents through their app. They can choose to share their medical data with the doctor when they go for check-up. Doctors can use the doctor app to view their patients records and if patient allows can share those records with his/her colleagues if required. I have to design the securities behind these exchanges in the most efficient manner possible.

Plan:

(1) Design the basic structure and features offered by the application.

For this I have chosen 2 live application offered by Apollo and roundglass to get the abstract structure of the app.

This will be done inline the current ehr standards given by https://www.nhp.gov.in/ehr_standards_mtl_mtl

(2) Study of various techniques used for designing the security features.

example: In case the application uses 3rd party servers for storing the records(like AWS etc) , I will explore techniques like access - control mechanisms , online encryption and homomorphic encryption.

For studying about these techniques, I will use the textbook and resources like https://www.cybrary.it/ . I will use research papers I have collected for the same if time permits.

(3) Propose a system to ensure security of the EHR. This solution will try to answer questions like:

(1) How will the access key be shared?

(2) Does doctor get to make its own copy of the records?

(3) How to add more information to the records? Who will have the access?

(4) What if patient later decides to take back the access?

(5) What happens in case the user forgets the password (secret key)?

Outcome: A system design for the security of Electronic Health Records.

REPORT:

Objective: To propose a secure system for usage of electronic Health Records according the below **problem Statement:**

There are 2 mobile applications - Doctor Application and Patient App. The users use the apps to maintain their electronic health records. Patients organize all their documents through their app. They can choose to share their medical data with the doctor when they go for check-up. Doctors can use the doctor app to view their patients records and if patient allows can share those records with his/her colleagues if required. I have to design the securities behind these exchanges in the most efficient manner possible.

The project also includes an attempt to explore Homomorphic encryption for machine learning in the healthcare domain.

Motivation: My Practice School – I was at a healthcare firm developing similar mobile applications as above. By working on them, I realized there are practically no laws regarding the handling of the Electronic Health Records. As announced by the Government of India, a new bill is in making to tackle this issue. Therefore, I am attempting to design a secure system which can be used in such use cases. The project also focuses on use of Homomorphic Encryption to perform Machine Learning on the encrypted health records.

Introduction:

The main aim of this project is to explore and design a security system for usage the Electronic Health Records and to perform Machine learning tasks on homomorphically encrypted data. Healthcare Industry is booming lately especially because of the technical revolution in the current decade.

In this world of machine learning, data has surpassed oil in terms of value, but Medical data is still considered out of reach. This is mainly because of the stiff HIPPA regulation and sensitivity of the medical data in general.

This project is an attempt to develop systems for Applications for the medical domain which involve sharing of EHR as frequently as in a clinic / hospital.

The final part of the project tries to perform machine learning tasks using homomorphically encrypted data.

Homomorphic Encryption:

Homomorphic encryption is a form of encryption that allows computation on ciphertexts, generating an encrypted result which, when decrypted, matches the result of the operations as if they had been performed on the plaintext.

Homomorphic encryption can be used for privacy-preserving outsourced storage and computation. This allows data to be encrypted and out-sourced to commercial cloud environments for processing, all while encrypted. In highly regulated industries, such as health care, homomorphic encryption can be used to enable new services by removing privacy barriers inhibiting data sharing. For example, predictive analytics in health care can be hard to apply due to medical data privacy concerns, but if the predictive analytics service provider can operate on encrypted data instead, these privacy concerns are diminished.

For the entirety of this project, I have used **a Paillier Crypto system** which is a partially homomorphic cryptosystem. Currently it supports basic math operations like addition, subtraction, multiplication and division with real numbers.

Paillier Crypto System:

The Paillier crypto system, invented by and named after Pascal Paillier in 1999, is a probabilistic asymmetric algorithm for public key cryptography. The problem of computing n-th residue classes is believed to be computationally difficult. The decisional composite residuality assumption is the intractability hypothesis upon which this cryptosystem is based**.**

App Design and the security system:

The system has 2 components mainly:

(1) Patient application: Installed on patient mobile, this application is an electronic version of the patient's health records. This application also holds the private key of the patient used in encrypting the patient's data.

(2) Doctors Application: Installed on the Doctors mobile, the application contains the patients list and records. The doctor requires a permission from the patient to see the their records (one time permission).

Some questions for the system:

Q1→ How is access key’s shared?

Ans → On the first visit, the doctor creates an account for the patient . Patient then accepts the request and Doctor can then access the records of the said patient.

Q2→ How to add more information to the records?

Ans → Every time patient gets prescription / lab test , the data is encrypted and then stored on the patient’s cloud. Only the personnel with the private keys can decrypt and see the original data (i.e. patient). Now whenever such a process occurs, patient has the option to broadcast the change or not. If the patient chooses to do , the Authentication codes / private key for that specific patient is also updates on the doctors application.

Q3→ What if patient later decides to take back the access?

Ans → As soon as patient changes the doctor or wants to deny further access, they can either stop broadcasting or deny the permission by removing the doctors from the broadcast list.

Q4 → How often will the data by synced?

And → On every update by the patient, data is synced. If authorized by patient , it is broadcasted on the cloud.

Files Included:

(1) example1.py

Use-Case : The hospital wants to know the average blood pressure of their patients. Now the data is homomorphically encrypted, but it can still be added and then decrypted later. This can be extended to dosage values (to check if the supply is sufficient or not) or detecting anomalies. (like epidemic outbreak because of higher demand of a particular drug)

a) This example takes the 10-day period Blood Pressure level of the patient in an encrypted form and then returns the Mean Blood pressure to the doctor.
b) The 10 values can be changes at the line 21 of the code.
c) run command: python example1.py
d) Here data is first encrypted by homomorphic encryption and then added without decrypting. The final average is then decrypted to know the actual average Blood pressure.

Sample Encrypted BP:

Blood Pressure of 80 is encrypted as :

18230554797424041192100285300827207433415701139024011327705148631181767248186732224107346138776982961842611035327628871623422722376751035500210433535409551546579249379097396282228904114478769322908824642861136204073540263958557050148289437718266910965198631647454216944224473556881653237178744009093781372847224446108709132178962909358755326549644489056853725106762863487021186246929078371377318900701813893521221599540828681960265826488841454635030893340625790945702169146473416329479842548511755022351263235095963605698711861418036787249259676325281301682069649272825463991272629771647211559494514174597724834734034765783487649599285842460598407357494281948736636129853484110464744572621803080825329236927789345200539290784006961489936974116189585390714783853686588319843183752708817156178176724619666955591897424104643562289847125686730334092225437766131452150467252980912 50

7147167204985624224547017382747424931836534548753837881604695410764755987
8010141988822917262913824458716198868036207093750822988950529520402979982
5340337791249320373676707259325638511396310974084298021110260037818270423
1931881612795349538999694439005283800150641959553752124030081333996651860
31638393693717909931240789553668148212926647822816505510191281

(2) example2.py

Use-case : This is how the patient data will look like. It takes the patient data as input, separates the Numerical data for homomorphic encryption and left data as normal encryption(**I have used AES and SHA256**.).

a) The Encrypted Data is stored in the file formdata.txt.
b) I have used **cryptodome library for encrypted non-PID based data** i.e. first name, last name etc. etc. directly taken from examples in the cryptodome documentation.
c) run command - python example2.py
d) Example:

Input

This example takes the normal patient form and encrypts the data

Please enter the required Details:

Please Enter your firstname:

lorem

Please Enter your lastname:

ipsum

Please Enter your age:

21

Please Enter your height(in cm):

175

Please Enter your Blood Pressure (Distolic):

120

Encypted Data is available in the file formdata.txt

Output:

{'firstname': 'seNlAopjxQmhxTRl1MelE6ZRheZ3h3W2UnVEwmI5tP0=',

'lastname':'uLkwhADkQ3pCuhu9jXlicxuR7PPjjAI+uAuLX0prVCc=',

'age':'17407183698015688702322986642451314609914573477233469048158609298273786

6899755688082544062268488692349323461844466263790251318399496737330276361196

6287231007566459432890937615621288324555816330079622643535259326107192854776

8493328394941555391283942482483964087236682212404713357511447517633213135302

4793357907480969488301382492067429435680643128578701210717837677919175437754

8416260452384140372257680965369803325013497035631171660885824004168726136168

3018566753982519132256527443428561331276372899513446153416082208878386755801

6608738134059185208680863320024215561258708156131867503859567640623559971649

7665062985836656207550382305370889612290829306433406477305545998767595495183

2018037002300150210978050926870213676915459050415165227415625084130140212989

6686479417843480472565255882365324582333996969539282535604113092828614698484

7704866767249929092790121195378168173365211092264809247346422925358233273018

3638819469583053820342051295384099116467103478223629331684614880132576729622

2198773940172954081591250201435078413449383243511378591155973906595069988295

9346680960385241739271289671279501436122007932645312497785388359874708841288

6914318042523271408558474323281977709358333592392145169345213513818903841261

7162356516915826939608',

'height':'4392489235779831099316310318138076140585251320288120069462090945537506
0125332587391200397501381223026639620697749373150194060270954538332804961178
8587178763611835381366026600099154128160991049199800206196756626129571684857
4274862625746031963179707697732133083402436665617024450550274873104085412856
3278722197482227508625463593434352922472034338217528898684009724574072523342
7918547501765014106341197213498833935503379827705364954420969480238263314030
6078049957688474429012119821497763080130096506777419952679955683543316188243
0061871513490049337261255365629968461331983099046294524097807865709152574317
0388568061610630883861427899791734317855465175376608088463847416090890189971
9261590541701996557374986819132915218363654894893883539916180779097335421470
7500418994180971295838708196778671109095101947661322178016326263835417146475
8500290383725719235012559257842449565469141893404294608919321262712495795754
1243443869204122457906500519976311576363393657607532391448308517120333711007
1575240180199582681482254961691050852567902681936592340263390136527621218459
8333256277469191452633666174662379850834122847471804840797655039832158563886
8844238776657821410058536602906843773007499153829524480174225215994059017443
27891281330717622301786',
'bloodPressure':'2624325751724666084441489577000435559139161416726972899337002531169838279767010368738808978142251009202168725636229332431311135104751109333
9725744406076837571572229836851208503307235555333127186068295715305572565677
2605915172627445581285348340695308632099333565710652925217340808307576417094
3579310097737894220435174929872152095976247389101736059068498522121626990274
3461920816854030203267757672410138314392025314938810335464586235962446707297
4657264704407081662284160082302064761667109918064674221039509220669584946901
8604533355785191893139786445657784345953176763888655625471670147895338233410

4245979686580731077896537627419210716227987868250507425193368448741752870226
5378154019483094658096192372625801693873747672149197760134970569331424518632
0649014351250918001708286667504440276344075180249534563914646056181469490512
4808773524759990167261026854431966806501835411166821870311190525093633402306
8521445330452285665046990592671171954941968993917226727717047409436819148178
6160011699863371575387551721016794049031346861658560600784955508402586643092
6640616478604240375380644602162206092877876543132540181515837587560637093735
0805740181715369009133494166388018225802955685764589059682479468995862945177
02008658611056643118759996021l'}

(3) Example3

UseCase: Partial Implementation for predicting Breast Cancer using knn(k-nearest neighbor) Classifier.

a) The given jupyter notebook has partial implementation of the knn classfier for the dataset.
b) The **implementation is partial because of lack of <= and > =** operations in the pallier cryptographic system.
c) The current implementation converts the given data in usable encrypted formats and have been tested for **basic statistical ratios like mean, average , variations etc.**
d) The encrypted data is stored in the file called encrypted_data.xlsx.
e) This has been tested in **another library called Microsoft seal** which has the same accuracy as the plaintext data.

Learning Outcome

(1) Concepts involving Homomorphic Encryption.

(2) Python Programming and some relevant libraries like pycyrptodome

(3) App Design Concepts and basis penetration testing concepts.

(4) Machine Learning – knn classifier

(5) Cryptographic techniques like AES and SHA256.

References:

(1) https://en.wikipedia.org/wiki/Paillier_cryptosystem

(2) https://pypi.org/project/pycryptodome/

(3) https://www.pycryptodome.org/en/latest/src/examples.html

(4) https://python-paillier.readthedocs.io/en/stable/phe.html

(5) https://www.nhp.gov.in/ehr_standards_mtl_mtl

(6) https://hal.archives-ouvertes.fr/tel-01918263/document

(7) https://www.youtube.com/playlist?list=PL-osiE80TeTt2d9bfVyTiXJA-UTHn6WwU

(8) https://play.google.com/store/apps/details?id=glass.round.cross.doctor&hl=en_IN

(9) https://play.google.com/store/apps/details?id=glass.round.reach&hl=en_IN

(10) https://www.microsoft.com/en-us/research/project/microsoft-seal/

(11) https://github.com/Lab41/PySEAL

(12) https://github.com/ibarrond/Pyfhel

(13) http://www.cs.tau.ac.il/~fiat/crypt07/papers/Pai99pai.pdf