Multi-user protocols with access control for computational privacy in public clouds

Sashank Dara∗ †
∗ Cisco Systems India Pvt Ltd,
† International Institute of Information Technology,
Bangalore, India

Abstract—Computational privacy is a property of cryptographic system that ensures the privacy of data being processed at an untrusted server. Fully Homomorphic Encryption Schemes (FHE) promise to provide such property. Contemporary FHE schemes are suited for applications that have single user and server. In reality many of the cloud applications involve multiple users with various degrees of trust and the server need not necessarily be aware of it too. We present a Complementary Key Pairs technique and protocols based on that to scale any generic FHE schemes to multi user scenarios. We also use such technique along with FHE to show how attribute based access control can be achieved while server being oblivious of the same. We analyze the protocols and their security. Our protocols don’t make any assumptions on how FHE scheme itself works.

I. INTRODUCTION

Cloud computing is well defined and standardized [1]. The major hindrance for cloud adoption is still computational privacy. It is a property of cryptographic system that ensures the privacy of data being processed at a remote untrusted server.

Informally in a cloud computing setup, Alice has inputs \{x_i\} to a function f to which she needs results, but lacks enough resources to compute the output. Sally, a service provider, has resources to execute f(x) and is willing to let Alice use them may be for a price. Similar to any Cloud Service Provider like Google’s Compute Engine, Amazon’s WebServices etc. Alice is concerned though that Sally would misuse the data. We assume Alice trusts Sally to give her the right answer of f(x_i) though.

To address such concern, one needs cryptographic techniques that allow arbitrary functions to be evaluated on encrypted data. So that Alice can send encrypted inputs \{x_i\} and Sally could perform functions on such encrypted data. Fully Homomorphic Encryption (FHE) schemes provide such functionality. They are proven to be practical by Gentry [2]. Current FHE schemes are highly inefficient and still not yet feasible [3].

Informally, FHE schemes allows one to perform fundamentally addition,multiplication over the cipher texts without ever decrypting the inputs. Apart from the keygen, encrypt, decrypt algorithms of Public Key Encryption (PKE) systems, FHE schemes consist of an additional algorithm evaluate. Evaluate method takes input as any arbitrary function f in boolean circuit form, encrypted inputs of f and executes it over encrypted data using the Public Key. The output is still in encrypted form.

\[ C_f = \text{evaluate}_{PK}(f, C_1, C_2, \ldots, C_n) \quad (I.1) \]

A. Problem

Adopting FHE schemes for cloud applications that involves just one user and server is straightforward the data can be encrypted using the Public key and outsourced to the server, the server can then perform operations using Evaluate method and Public key and return the results that are still in encrypted form. The user then decrypts the results using Private Key.

But most of the cloud applications involve multiple users collaboratively utilizing a service. Encrypting data with their individual Public keys would not result in encrypted data that can be operated upon by them collectively.

B. Straw man Solution

In order to make the encrypted data coherent even after encryptions by multiple users, all the users can have same key pair for encryption and decryption. But this would lose accountability and causes single point of failure. In addition collaboration based on different degrees of trust that needs access control measures cannot be enforced.

So we precisely need techniques to keep the data coherent even encrypted by multiple users but still each user is accountable for their operations. Additional access control measures also should be enforced easily.

C. Key Contributions

1) We introduce Complementary Key Pairs technique that involves two distinct key pairs that complement each other to achieve the desired functionality.
2) We design protocols for scaling FHE schemes for multi user scenarios and analyze their security using such technique.
3) We also show how Server Oblivious - Attribute Based Access Control (SO-ABAC) can be achieved over encrypted data using such technique.

II. PRIOR ART

Current popularly published FHE schemes are suited for single user utilizing some cloud service, they are not for multiuser.

Xiao et al proposed protocols for Multi User systems [4], that are based on symmetric Homomorphic Encryption scheme
that could evaluate functions only on polynomials. Also their protocols are tightly coupled with their scheme.

Also set of literature exists for achieving multi party computation using threshold homomorphic encryption \cite{5,6} and also multikey homomorphic encryption \cite{7}. These techniques require few of the users to collaborate to decrypt the final result, which is not reasonable to assume especially in the cloud setup.

Non interactive Verifiable Computation \cite{8} is another concept introduced to verify the computations when they are outsourced. Such techniques have been adopted for multiuser scenario too by Choi et al \cite{9}. The multi user protocols introduced here encrypt the same input under multiple identities and stored as a map, this would not scale when the number of users or the input data increases in size.

Dijk and Juels argue FHE alone cannot solve cloud privacy problems\cite{10}. They prove that, realization of private multi-client scenario is equivalent to impossibility results of program obfuscation. They assume, in multi client scenario the encryption is done with different Public keys. They also state that functionality beyond FHE is needed like Access Control over cipher texts and Re-encryption to achieve such private multi-client scenarios.

Their impossibility proof makes an assumption that isn’t true in our case. In our protocols, even in multi client scenario the encryption of data is done by single Public key and authentication of users is done with different individual key pair. Also we prove (by construction) that such additional functionality is purely application of FHE. Our protocols are counter examples to their core results.

Access control over encrypted data is achieved using Attribute Based Encryption schemes \cite{11,12}. Such techniques in combination with Proxy Reencryption schemes were adopted to cloud setting \cite{13}. Such adaptations are complex in nature for practical implementation. Also Receiver Oblivious Attribute Based Access Control has been achieved through attribute based encryption \cite{14} and this does not fit in cloud setup.

III. PRELIMINARIES

A. Users

In all of the below protocols, Alice, Bob, Carol are typical Cloud Service Consumers, can be an individual or an enterprise. Sally is a typical Cloud Service Provider like Google, Amazon or any other provider.

B. Services

The service can be based on any of the Infrastructure-as-Service (IaaS), Platform-as-Service (PaaS), Software-as-Service (SaaS) delivery methods \cite{11}.

C. Adversaries

In real world multiple adversaries exist, the cloud platform itself, somebody compromising cloud platform, neighbors sharing the platform can be malicious. Such differentiation is not important though if proper computational privacy measures are in place. Abstractly such adversaries can be modeled as single adversary, Mallory.

D. Attacks

The goal of Mallory is to get unauthorized access to plain text and/or to corrupt the cipher text. Mallory can compromise the cloud platform itself or exploit a flaw in the protocol to achieve such goal.

E. Outline

We start with a trivial protocol for Alice to outsource her data \cite{11} to Sally while ensuring computational privacy of the same. We introduce the concept of Complementary Key Pairs. We then present protocol for Alice to collaboratively work with Bob on shared data using such technique. Using this protocol both Alice, Bob can execute arbitrary operations on encrypted data. Further we present a protocol that allows Sally to enforce oblivious access control over Alice’s data when multiple users are present. They are analyzed for possible attacks and vulnerabilities.

F. Basic Assumptions

- Alice, Bob and Sally know each other’s public keys \cite{11}
- Alice, Bob and Sally mutually authenticate each other
- Sally knows legible partners of Alice through some initial configuration.
- All the communications take place on a secure channel

Note: Protocols stated here are for illustration purposes, mainly to drive the point. Security is discussed only for the functionality they represent to solve in this context. Conventions followed in protocols are detailed in Table \cite{11}

IV. BASIC PROTOCOL

In this protocol, we consider the basic scenario where only two parties are involved, Alice the user and Sally the service provider.

A. Summary

Alice encrypts the data using her Public Key and uploads (i.e outsources) it to Sally. At any later point of time, Alice requests various operations on the encrypted data using API’s provided by Sally, Sally performs the operations on the encrypted data using FHE Scheme’s Evaluate method and Public Key of Alice. Sally returns the results to Alice. Alice decrypts the results locally using her Private Key.

The detail steps of protocol are mentioned in Table \cite{11}

B. Security

In this simple setup, If cloud platform itself is compromised by Mallory, FHE schemes guarantee the computational privacy of Alice’s data. FHE schemes also guarantee that accidental or intentional sharing of encrypted data with other users or service provider’s would not leak anything.

\footnote{In communications the abstraction of plain text is message, but in computations data fits well.}

\footnote{In all the protocols, such keys are given offline or got from trusted server or any other safer means}
Each of these users have their own Public, Private key pairs. Encrypting the data with their respective Public keys will not result in encrypted data that can be collectively operated upon. At the same time if same key pair is shared among and used by all the users then it would cause single point failure if the keys are compromised. Also there would not be any accountability on who made the change exactly if every one uses same Public Keys.

A. Complementary Key Pairs

Two distinct types of key pairs are used together through out the protocols. The key pairs complement each other to achieve the required functionality.

1) Authentication key pair auth-Private, auth-Public Keys for Users (Alice, Bob) to authenticate themselves with Server (Sally). All the users and server have their own respective auth pair.

2) Evaluation key pair eval-Private, eval-Public keys for Users to execute arbitrary operations with Server. All the legible users (partner’s of Alice) would have one common Eval key pair. Server knows eval-Public key only.

This is by design principle Separation of Concerns. Each key pair is used for separate concern aka functionality.

B. Summary

Alice, her partners and Sally generate auth-Private, auth-Public Keys, this can be either through traditional PKE schemes or FHE. Alice also generates eval-Private, eval-Public keys using FHE scheme. Alice ensures same Eval key pair to be available with all her partners. Such sharing of Eval key pair is reasonable and similar to having a pre shared secret in Symmetric key crypto systems.

All the users authenticate themselves using their individual auth key pairs while requesting any operations. The data is encrypted with eval-Public key when any of the users upload/write it to the cloud and results are decrypted with eval-Private key locally at the users. Further the actual execution of the operations are carried out by server using eval-Public and Evaluate method of FHE scheme. Since the data is encrypted under same eval-Public key by everyone, the encryptions by multiple users are coherent.

The detail steps of protocol are mentioned in Table III. Bob’s run of the protocol would be similar, by using his authentication keys, without loss of generality.

C. Security

All the users use their own auth-keys used for authentication purposes. The actual encryption of the data is done using eval-Public key. Since the encryption is done under single key by all the users, its easy for the server to run Evaluate method of FHE scheme. Similarly all the users (except the server) have eval-Private key for decrypting the results locally.

Mallory cannot request for arbitrary operations over data shared by Alice, Bob, such requests would be dropped while verifying the signature since Sally knows the legible users. If Mallory some how compromises any user’s auth-Private
users apart from authorizing granular access control. In many real world scenarios, a user would authorize partners to obtain results of only certain operations. Also, users with different access control privileges. In many real world scenarios, a user would authorize partners to obtain results as that would need eval-Private key alone will not allow Mallory to perform operations on the data since authentication is done by auth key pair.

The security of the system depends though on adequate measures for safer key distribution and authentication mechanisms.

D. Example Use cases

In the below cases Eval key pair can be shared among Alice and her partners.

- An Enterprise has its branch offices located at various different locations. All the branches collectively need to work over data and applications with remote cloud server.
- An End user wants to store his personal data with cloud server. It is very likely to share confidential data with few other users (like family members).

VI. Multi-user Constrained Secret Protocol

A much richer context is where Alice authorizes her partner users with different access control privileges. In many of real world scenarios, a user would authorize partners to obtain results of only certain operations. Also Alice would like to have control on addition, deletion, revocation etc on the partner users apart from authorizing granular access control.

Informally, Access control is authorizing subjects (users, applications or processes) with certain levels of access to various resources (files, devices etc) and enforcing such authorization when the resources are accessed. A generalized model of all the traditional ones is Attribute Based Access Control (ABAC)\(^5\). ABAC is an excellent choice for web services like cloud computing due its feature richness\(^\text{16}\).

We give a quick introduction to ABAC model. We continue to use the complementary key pairs technique with additional restrictions for access control. Although all the legible users have the eval-Private key to decrypt the results they are further constrained to perform only certain operations, thus the name constrained secret protocol. We first show how access control can be achieved on encrypted data using FHE schemes and then use those techniques in the protocol.

A. Attribute Based Access Control (ABAC)

In ABAC model, permissions are defined based any characteristics known as attributes. The attributes for subjects and resources can be identifiers, names, title etc. ABAC also has special kind of environmental attributes like current date, security levels etc. A policy rule defined as a boolean function of attributes to decide if a subject s can access a resource r in an environment e as below.

\[
\text{canAccess} = f(\text{ATTR}(s), \text{ATTR}(r), \text{ATTR}(e)) \quad \text{(VI.1)}
\]

An Administrator defines such attributes and also decides such policy rules. They are stored in a Policy Rule Base(PRB). A Policy Enforcement Point (PEP) is responsible for receiving requests for authorization decisions and enforcing them. PEP in turn takes help of Policy Decision Point (PDP) to evaluate the validity of authorization request by processing Policy Rules.

In our current cloud setup, The subjects are Alice, Bob. Encrypted data fine grained as files or database records are resources. Alice as the data owner takes the role of an administrator, defines on the attributes for partner users and resources. Alice also defines the policy rules she wants to grant as privileges for her partners. Alice outsources such PRB to Sally the server. Sally acts as both Policy Enforcement and Decision Point. Also add, delete, revoke of partner users can be achieved using the PRB. For example, absence of privileges for a particular user in PRB can be considered as the user is deleted or revoked.

In Cloud setting, Policy Rule Base(PRB) in plain form, at the server, can leak information on the underlying data. For example, if a policy rule as "Carole can write to a file xyz.jpg", would leak information. For this reason, the PRB also need to be encrypted so that server is oblivious of such details.

Server Oblivious - Attribute Based Access Control(SO-ABAC) can be achieved using FHE schemes together with Complementary Key Pairs technique introduced in previous protocol. Attributes can be defined for users identified by their respective auth-Public Keys. The resource identifiers can be encrypted using eval-Public Key of FHE scheme. Enforcement of such access privilege can be done, obliviously a.k.a blind-foldedly, using Evaluate method for any user requesting access for a resource.

\(^3\)In reality attribute administrator and policy administrator can be different too

| TABLE III. Multi user Shared Secret Protocol Steps |
|-----------------------------------------------|
| Data Preparation |
| 1 A := data \(\equiv encrypt_{eval-PK}(data)\) |
| 2 A \rightarrow S := sign_{auth-SKa}(data) |
| 3 S := verifySig\(_{auth-PK_a}(sign_{auth-SKa})\) |
| Protocol Steps |
| 1 A := x\(\equiv encrypt_{eval-PK}(x)\) |
| 2 A \rightarrow S := sign_{auth-SKa}(\text{func}\cdot x) |
| 3 S := verifySig\(_{auth-PK_a}(sign_{auth-SKa})\) |
| 4 S := y\(\equiv evaluate_{eval-PK}((\text{func}\cdot x)\cdot data)\) |
| 5 S \rightarrow A := sign_{auth-SKa}(y) |
| 6 A := verifySig\(_{auth-PK_a}(sign_{auth-SKa})\) |
| 7 A := y = decrypt_{eval-SKa}(y) |
The attributes for resource and environment depend on the identification of the user as partner user requests operations. Grants access to users as per encrypted PRB. The previous protocol defines PRB. Sally as Administrator configures PRB and subsequently updates the PRB with identification attribute of Bob. When an user requests certain operations, he/she needs to send certain attributes. The attributes of a user Bob used for identification is auth-PK\textsubscript{b}, there can be other attributes too. The attributes for resource and environment depends on the operation being requested lets assume they are r.e. All the attributes and PRB are encrypted using eval-Public key.

\[ \text{ATTR}(1) = \text{encrypt}_{\text{eval-PK}}(a_1, \ldots, a_n) \quad a_i \in \{s, r, e\} \] (VI.2a)

\[ \text{ATTR}(s) = \text{auth-PK}_{b} \] (VI.2b)

\[ \text{ATTR}(s) = \text{encrypt}_{\text{eval-PK}}(\text{auth-PK}_{b}) \] (VI.2c)

\[ \text{PRB} = \text{encrypt}_{\text{eval-PK}}(\text{PRB}) \] (VI.2d)

Policy enforcement decision can be done at the server using the \textit{Evaluate} method of FHE scheme as shown below, this is later used in the protocol. Assume \textit{canAccess} (VI.1) is a boolean circuit of ABAC model.

\[
\text{verifyAccess}_{\text{eval-PK}}(\text{func}, \text{ATTR}(1)) = \text{evaluate}_{\text{eval-PK}}(\text{canAccess}, \text{func}, \text{ATTR}(1), \text{PRB})
\] (VI.3)

The detail steps of protocol are mentioned in Table [V] Alice's run of the protocol, as user, would be similar, by using his authentication keys, without loss of generality.

### C. Rekeying, Revocation and ReEncryption

If Bob's auth key pair is compromised, He generates a new key pair and sends the auth-Public key to Bob. Alice needs to update the PRB with identification attribute of Bob containing the new public key. If Alice needs to revoke a particular user, she creates and encrypts such a policy rule and sends a update PRB request to Sally.

If eval-Private key is compromised, Alice generates a new eval' key pair and distributes it to all the partner users. She lets Sally know the new public key eval-PK\textsuperscript{'} key and requests a re-encryption of data. Sally executes the evaluate method with the encryption circuit and new public key as below

\[ \text{reEncryption} : \text{evaluate}_{\text{eval-PK}}(\text{encrypt, eval-PK'}, \text{data}) \]

### D. Security

If Mallory is not a legible partner but tries to access resources that are forbidden, such requests are first thwarted while verifying signature itself, since its assumed Sally knows all legible users through some initial configuration. If Mallory is legible partner but tries to access resources that are forbidden, such requests are thwarted while verifying the access control privileges. If Mallory tries to overwrite the PRB itself, such requests are thwarted too since such administrative operations are allowed by Sally only from Alice.

A risk exists though, If Mallory compromises the cloud platform itself and corrupts the PRB or bypasses the access check, this could result in a complete denial of service. But such \textit{Single Point of Failure} is inherent to cloud computing model itself, there is a total denial of service to user due to poor internet connectivity or if cloud service is down (either due to technical problems or attacks).

### E. Example Use cases

- \textit{Privacy of Health Records} Hospitals can decide access control on their data based on who their partners
are for example insurance, doctors, researchers etc. Server can be completely blind about the data being processed and also the access privileges for resources.

- **Private Social Networking** Each user can authorize, who (friends, family, everyone etc) can access their data (like photos, blogs etc) and this can be done without the server ever knowing it.

VII. Future Work

We have assumed, in our protocols, the cloud server would be honest in performing the computations correctly, in real world such assumptions cannot be made. Accidentally or malicious execution of arbitrary computations over encrypted data might effect the results and/or render data useless too. For this reason, *verifiability of the computations* is very important. Current verifiability techniques are nascent [8][17][18]. Enhancing such techniques and tailoring them into the above protocols would be for further work. Rigorous proofs of security protocols for computational privacy can be achieved when *verifiability* of computations is integrated with them.

Current FHE schemes are highly inefficient and theoretical to conduct experiments and provide results. Also Cloud Computational Privacy protocols being nascent, there aren’t any theoretical frameworks to best capture their security requirements and conduct rigorous analysis. The current popularly used Universal Composability framework has its own limitations [19] to be applied directly to cloud protocols.

VIII. Conclusions

We introduced *complementary key pairs* technique for multi user scenarios. Three protocols which cater to gamut of use cases of cloud computing identified by the community [20] are presented in our work. Also we showed that FHE schemes for single user can be scaled to multi user scenario too and no special variants of FHE schemes are required for the same. These protocols make no assumptions on underlying schemes, this gives them flexibility of adapting them to any FHE schemes. We also showed how oblivious (blind-folded) *access control* over encrypted data can be achieved using FHE schemes.

IX. Acknowledgments

I would like to thank Cisco Systems for supporting this work. I would like to thank my research advisor Dr V.N. Muralidhara for all the insightful discussions and brainstorming. I would like to thank my colleague and Technical leader Scott Fluhrer, a brilliant cryptographer in Cisco for providing valuable suggestions for this paper. I would like to thank Cisco Fellow Dr David McGrew for all the opportunities he created for me to learn cryptography and cloud security in detail. I would also like to thank crypto.stackexchange community for having many insightful discussions around this topic.

REFERENCES

[1] P. Mell and T. Grance, “The nist definition of cloud computing, special publication 800-145,” US Department of Commerce, Gaithersburg, MD, 2011.

[2] C. Gentry, “A fully homomorphic encryption scheme,” Ph.D. dissertation, Stanford University, 2009.

[3] V. Vaikuntanathan, “Computing blindfolded: New developments in fully homomorphic encryption,” in Foundations of Computer Science (FOCS), 2011 IEEE 52nd Annual Symposium on. IEEE, 2011, pp. 5–16.

[4] L. Xiao, O. Bastani, and L-L. Yen, “An efficient homomorphic encryption protocol for multi-user systems.”

[5] R. Cramer, I. Damgård, and J. B. Nielsen, *Multiparty computation from threshold homomorphic encryption*. Springer, 2001.

[6] S. Myers, M. Sergi, and A. Shelat, “Threshold fully homomorphic encryption and secure computation,” *eprint*, vol. 454, p. 2011, 2011.

[7] A. López-Alt, E. Tromer, and V. Vaikuntanathan, “On-the-fly multiparty computation on the cloud via multikey fully homomorphic encryption,” in *Proceedings of the 44th symposium on Theory of Computing*. ACM, 2012, pp. 1219–1234.

[8] R. Gennaro, C. Gentry, and B. Parno, “Non-interactive verifiable computing: Outsourcing computation to untrusted workers,” in *Advances in Cryptology–CRYPTO 2010*. Springer, 2010, pp. 465–482.

[9] S. G. Choi, J. Katz, R. Kumaresan, and C. Cid, “Multi-user non-interactive verifiable computation.”

[10] M. Van Dijk and A. Juels, “On the impossibility of cryptography alone for privacy-preserving cloud computing,” in *Proceedings of the 5th USENIX conference on Hot topics in security*. USENIX Association, 2010, pp. 1–8.

[11] V. Goyal, O. Pandey, A. Sahai, and B. Waters, “Attribute-based encryption for fine-grained access control of encrypted data,” in *Proceedings of the 13th ACM conference on Computer and communications security*. ACM, 2006, pp. 89–98.

[12] J. Bethencourt, A. Sahai, and B. Waters, “Ciphertext-policy attribute-based encryption,” in *Security and Privacy, 2007. SP’07. IEEE Symposium on*. IEEE, 2007, pp. 321–334.

[13] S. Yu, C. Wang, K. Ren, and W. Lou, “Achieving secure, scalable, and fine-grained data access control in cloud computing,” in *INFOCOM, 2010 Proceedings IEEE*. IEEE, 2010, pp. 1–9.

[14] J. Han, W. Susilo, Y. Mu, and J. Yan, “Attribute-based oblivious access control,” *The Computer Journal*, vol. 55, no. 10, pp. 1202–1215, 2012.

[15] X. Jin, R. Krishnan, and R. Sandhu, “A unified attribute-based access control model covering dac, mac and rbac,” in *Data and Applications Security and Privacy XXVI*. Springer, 2012, pp. 41–55.

[16] E. Yuan and J. Tong, “Attributed based access control (abac) for web services,” in *Web Services, 2005. ICWS 2005. Proceedings. 2005 IEEE International Conference on*. IEEE, 2005.

[17] S. Benabbas, R. Gennaro, and Y. Vahlis, “Verifiable delegation of computation over large datasets,” in *Advances in Cryptology–CRYPTO 2011*. Springer, 2011, pp. 111–131.

[18] K.-M. Chung, Y. Kalai, and S. Vadhan, “Improved delegation of computation using fully homomorphic encryption,” in *Advances in Cryptology–CRYPTO 2010*. Springer, 2010, pp. 483–501.

[19] R. Canetti, E. Kushilevitz, and Y. Lindell, “On the limitations of universally composable two-party computation without set-up assumptions,” in *Advances in Cryptology–EUROCRYPT 2003*. Springer, 2003, pp. 68–86.

[20] community, “Cloud computing use cases,” *A White Paper Produced by the Cloud Computing Use cases, 2010*, [http://bit.ly/gixdL7](http://bit.ly/gixdL7).