A Privacy Preserving of Composite Private/Public Key in Cloud Servers

O Sri Nagesh
PhD Scholar, Department of CSE, Lingaya's University, Faridabad

ABSTRACT *Security is a term used to provide secrecy of data from the illegal entries. It is used to prevent a user that he/she should not have access to. It is a two step process. The security system in the first step identifies who the user is by requiring the user to submit some sort of identification. This is called* ***authentication****, and it means that the system is trying to find out who you are. Once the system identifies who you are, in the next step the system has to determine that you should permitted to access the recourse. This part of the process is called* ***authorization****, and it means that the system is checking to see if you have privileges to perform a certain action. OAuth and OpenID are the authentication schemes are used for authentication but they also have privacy protection problems. Here we are proposing composite private key over cloud system which is protected by splitting key into different servers which combines to form a single composite key. In this approach we are having third party key servers used public key cryptography to save public/private keys of their clients. We are using ring signatures by forming the network into ring and public keys are shared from that ring structures. We are proposing this composite private/public key pairs for privacy preserving in the facebook and other social network. We also propose homomorphic algorithm to prevent the compromised key server problem with the attacker.*

Keywords –*Anonymity, Anonymous communication, Authentication, online social networks, Homomorphic encryption*

I INTRODUCTION

Face book, LinkedIn and other social networks got more popularity now a days and they must preserve the privacy of a user and they must have the capability of retaining the user's online identity. Social networks are also responsible for providing privacy for the users. The cross-site authentication protocols such as OAuth [1] and OpenID [2] are used for providing online identity across many sites. We found lot of attackers can exploit vulnerabilities in clouds and utilize cloud system resources to deploy attacks. User's online activity may be tracked by the attacker so privacy problem may occur for his/her data. We are proposing a novel approach for privacy preservation of a user over vulnerable cloud network. In this approach we are proposing composite public/private keys instead of one single public/private key. This composite public/private key generated by the user is divided into multiple small keys and distributed among the different key servers. Users can use these key divisions in privacy preserving, anonymous cryptographic protocols, through the use of ring signatures [3, 4]. We are using secured cloud system. In this system cloud of key servers that collectively assign key parts to each social network identity. The key distribution is necessary because if a server with single key compromises with attacker reveals the privacy of secret key. If the key is composite and split among multiple key servers any one server is genuine the privacy will be preserved. We are proposing a new system to split the key into multiple parts and each part encrypts with homomorphic algorithm and then they are placed in the key servers. This is because if all the servers compromise then also the privacy of private key must be preserved. We are only proposing to encrypt private keys of each client. There is no need to encrypt public key using homomorphic encryption. Each key server stores encrypted key part which can be decrypted only by the client of his private key.

Privacy preservation is a significant research issue in social networking. The social networking platforms are extended into the mobile environment, users are increasing more require more extensive privacy-preservation because they are unfamiliar with the neighbors in close vicinity who may store, and correlate their personal information at different time periods and locations. Once the personal information is correlated to the location information, the behavior of users will be completely disclosed to the public. The content-

sharing applications, all of which provide no feedback or control mechanisms to users and may cause inappropriate location and identity information disclosure. To overcome the privacy violation in MSNs, many privacy enhancing techniques have been adopted into the MSN applications.

We are assuming about the adversary of our system. We are assuming the client can connect to the anonymous network like Tor [6], which is having secured connection like SSL (Secured Socket Layer). If all the key servers are dishonest it is essential to protect our private key by using homomorphic encryption. Previously anytrust model [5] is used. We are assuming in Anytrust model that atleast any one server must be genuine, and then the privacy of private key is preserved. If all the servers are compromised the security of private key is violated. And also we need to assure that all key servers must return the same key for the same request at different times. They must not return different keys for same key request. The key server must not be ambiguous. Key server can see the IP addresses of the clients connect with the server.

II SYSTEM ARCHITECTURE

The idea behind the architecture in figure 1 is in order to log into a social network like twitter or face book the social network supplies OAuth token for us. The client connects to key servers cloud through secure network like SSL which is secured encrypted end to end connection between client and each key server. The purpose of OAuth token is, In order for the client to access resources; it first has to obtain permission from the resource owner. This permission is expressed in the form of a token and matching shared-secret. The purpose of the token is to make it unnecessary for the resource owner to share its credentials with the client. Unlike the resource owner credentials, tokens can be issued with a restricted scope and limited life time, and revoked independently. Anonymous network like Tor is used for connecting to key servers. Then the client contacts the key server with its authentication using OAuth token. After the authentication is verified by the key server it provides the client with public/private keys. When the client wants to send some data to any client he/she takes all the keys from all key servers and decrypt using homomorphic decryption inorder to get his/her private key. Using this decrypted private key he/she can send message by encrypting that message and sends to the sender. All public/private keys partly are stored in the key server's data base. These parts are taken to get a single public/private key. The private keys are further secured by applying homomorphic algorithm.

Public-key encryption, in which a message is encrypted with a recipient's public key. The message cannot be decrypted by anyone who does not possess the matching private key, who is thus presumed to be the owner of that key and the person associated with the public key. This is used in an attempt to ensure confidentiality.

Digital signatures, in which a message is signed with the sender's private key and can be verified by anyone who has access to the sender's public key. This verification proves that the sender had access to the private key, and therefore is likely to be the person associated with the public key. This also ensures that the message has not been tampered, as any manipulation of the message will result in changes to the encoded message digest, which otherwise remains unchanged between the sender and receiver.

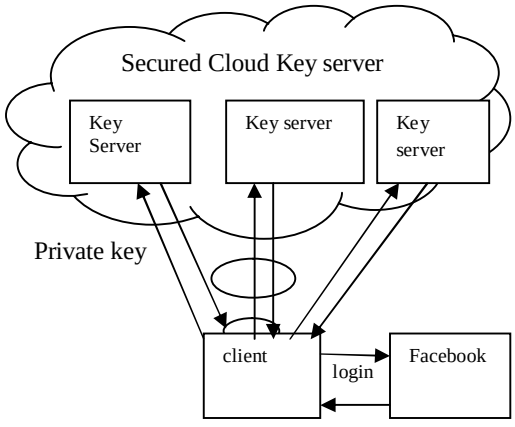

Figure 1 Authentication token(OAuth)

Each time a user connects to social network an OAuth token is provided to the user for getting his private key. There is no need for OAuth token for getting public key of any user. The homomorphic algorithm is used to encrypt the client's private key by him/her and placed in the key server in order to get more secured private keys when the key servers are compromised by the attackers. There is no need of Homomorphic algorithm for public keys because public keys are used only for encryption not for decryption. The initiator client simply asks the social network about the identity information of other responder client's public key parts from corresponding key servers. After initiator got the public key parts from the corresponding key server's then client combines these parts to get composite public key. Using the responder public key the initiator encrypts the information which he wants to send and sends to the responder. The responder decrypts the information sends by the initiator by using his private key taken from key servers. After using OAuth token from face book or other social network the initiator or client gets his/her own private key. Then the client constructs a ring signature [3, 4] with all other network identities acting as an explicit anonymous set. The trusted third party can verify the ring signature using public keys, that the signature is generated by any one client in the ring. It is not possible to determine who exactly created the signature. This ring signature generally forms by users from a meaningful group such as a company with some members that the data exchange must be provided secrecy among them who are in the ring network. The users are using this ring signatures a trust worthy privacy preserving structures.

2.1 Homomorphic Encryption

Homomorphic encryption is the conversion of data into ciphertext that can be analyzed and worked with as if it were still in its original form. Homomorphic encryptions allow complex mathematical operations to be performed on encrypted data without compromising the encryption. In mathematics, homomorphic describes the transformation of one data set into another while preserving relationships between elements in both sets. The term is derived from the Greek words for "same structure." Because the data in a homomorphic encryption scheme retains the same structure, identical mathematical operations -- whether they are performed on encrypted or decrypted data -- will yield equivalent results. Homomorphic encryption is expected to play an important part in cloud computing, allowing companies to store encrypted data in a public cloud and take advantage of the cloud provider's analytic services.

In our system partial private key generated by key server is encrypted by using the client system by applying homomorphic algorithm and only client can decrypt his private key part from each server. If all the key servers are compromised then the private key is secured by this homomorphic algorithm applied by the individual client. Only private keys are applied with homomorphic algorithm but not the public keys. If we consider $\pi_1 \ldots \pi_t$ which are parts of private keys homomorphic algorithm allows anyone (not only the key-holder) but they cannot understand the private key. This is done by encrypting each part of private key by $E(\pi_{1),..}E(\pi_t)$ then by using a function f to get a cipher text that encrypts f $(E(\pi_{1),..}E(\pi_t))$. Then all $E(\pi_1) \ldots E(\pi_t)$ are distributed to the key servers. Whenever the client needs his/her private key he/she gets all the keys $E(\pi_1) \ldots E(\pi_t)$ and applies f $(E(\pi_{1),..}E(\pi_t))$ to decrypt the private key.

2.2 Secured Cloud Key Server Model

In Crypto-Book [7] relies on a decentralized client/server model called anytrust [5]. In this model, we believe that any one key server is genuine. We don't bother which one is honest. All but that one server must be genuine is enough for privacy preservation. If all the servers compromise then privacy is not preserved. Under Crypto-Book threat model it is assumed that at least one key server is honest, and other key servers may be compromised with attackers by disclosing the client's private key. For each client, each key server i generates a private key part k_i. This k_i The client uses a combining function f that takes the private key parts k_0, $k_1 \ldots\ldots k_n$ and combines them into a composite private key $k_c = f(k_0, k_1 \ldots\ldots k_n)$. We

choose the combining function f such that all n key parts k_i are required to calculate k_c, and given any n-1 key parts, it is cryptographically infeasible to calculate or learn any significant information about k_c

Our enhancement is to encrypt each part of key by using homomorphic algorithm encrypt $E(k_{0)...}E(k_{n)}$ and then send them to appropriate key server. If all the key servers are compromised then also the encrypted key set is not revealed. Security will be enhanced.

Decrption is done by getting all the keys apply homomorphic decryption on each part and apply function f as shown $f(D(k_0)….D(k_n))$.

2.3 Security in getting Keys

Getting the keys securely from the cloud key servers is done by getting each part of the key and decrypting by using homomorphic decryption and combine all parts and apply to a function f to get his/her composite private key. It is shown in Figure 1**.** Inorder to protect the privacy of key servers from malicious users we connect the key servers through anonymous networks like Tor [6]. The anonymity network provides secure socket layer (SSL) connection to key servers. The client request for facebook or other social network request which the social network provides a token called OAuth, by using this token the client can obtain his/her private key parts from key servers. This is done securely by SSL connection between client and key server. When the client request for private key is authenticated by key server it supplies the encrypted private key part. This part is decrypted by homomorphic algorithm to retain original private key part. On receipt of a private key request and OAuth token, the key server connects to the social networking provider using the OAuth token, to verify the user's identity and obtain the user's unique social networking username. On receipt of all private key parts the client then combines these parts together into a composite key using a combining function.

2.4 Key Requester privacy

The key server knows who created the ring signature in the network. If it compromises it creates intersection attacks [8, 9, 10]. When key server is compromised it allows an attacker to see who requested their private keys. After ring signatures are generated by their clients and used, the attacker is able to calculate who created a ring signature. If we want to hide the details of the client who initiates the key request we are proposing an anonymous key pickup protocol that helps to protect client anonymity in the face of such intersection attacks. For example if X wants his/her private key part from the cloud key server, but X does not want to reveal about his/her key request to the other network in the cloud. X will not connect to social network directly instead of requesting his/her key request he/she sends some mail id's including his/her mail id to social network like face book. Then the face book sends the encrypted private key part using symmetric key through email. Then X decrypts his/her private key by decrypting it., as shown in Figure 2.

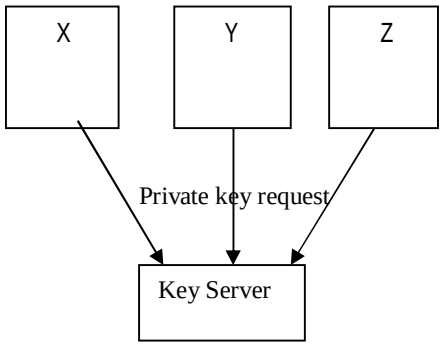

Figure 2 Anonymous key requests

The server sends over the secure connection to X the symmetric encryption key along with instructions on how to decrypt her private key. Emails reaches to the addressee only, which contains his/her own private key in the encrypted form. He only can decrypt by using some symmetric key generated by the key server. In

order to avoid too many messages to client's emails it is necessary to restrict the number. If X's private key was encrypted, her email provider or other attacker compromises his/her email or intercepts does not have access to it. If the server received an anonymous request from multiple clients and then sent out multiple private keys to multiple emails, the server does not know who in the end was able to decrypt and use their private key, in this way protecting X against intersection attacks from a compromised server. It is necessary that X needs to carry out this scheme only once per key server: e.g., if X is requesting five key servers, he/she need only do it for five times. X can then construct her private key and keep it saved for future use so as to avoid having to participate in anonymous key distribution again. In addition to email, this protocol could also be deployed over other communication channels such as social network messaging or SMS text messaging.

2.5 Digital(Ring)signatures

The ring signature is a popular way to concern about authentication of the user who forms a group. It is a signature that can be done by any person of a group of persons form a ring structure. Then a message signed with a ring signature is authorized by somebody in a particular assembly of community. It is difficult to determine who signed' to produce the signature. So privacy is also preserved in this data structure. Ring signatures [3, 4] developed by using group signatures [3, 4], the message signed with ring signature is authorized by some person of a different group, without disclosing which person's private key was utilized to generate the signature. These signatures are generated on an ad hoc basis using random sets of public keys. There is no need to know the owners of those public keys they are being conscripted into a secrecy set for signing purposes. We use these ring signatures to validate with third party servers. The third party server recognizes that the person is a member of a particular group of users, and the user identity is kept in secret that who created the signature, third party server doesn't know who created the signature. In Rivest et al.'s ring signatures it was understood that all ring members have an RSA public/private keypair. The person who sign's must get the public keys of all persons to use in the secrecy set, By using his/her own private key with the public keys of other persons in the ring generate ring signature. By using these kind of ring signature the malicious user or hacker not able to decrypt the originator who signed.RSA algorithm does not support key splitting, so we need to use a single trusted third party key server. Linkable ring signatures (LRS) [11]. Linkable ring signature is an alternative method such that two signatures can be linked if and only if they were signed by the same person. We can also use DSA style [12] keys here key splitting to multiple key servers are possible. Linkable ring signatures can determine from given two signatures, a trusted third party can resolve that they were produced by the same or different persons of the ring. Linkable ring signatures are ring signatures, but with added link ability such signatures allow anyone to determine if they are signed by the same group member (i.e. they are linked). If a user signs only once on behalf of a group, he still enjoys anonymity similar to that in conventional ring signature schemes. If the user signs multiple times, anyone can tell that these signatures have been generated by the same group member.

Ring signature is a group-oriented signature in which the signer can spontaneously form a group and generate a signature such that the verifier is convinced the signature was generated by one member of the group and yet does not know who actually signed. Linkable ring signature is a variant such that two signatures can be linked if and only if they were signed by the same person.

We use DSA style because and LRS which supports private key splitting and placed in multiple servers. In our architecture we use DSA while LRS will not support forward secrecy. Forward secrecy, in computer security and cryptography is the property which prevents an attacker who has recorded past communications from discovering the identities of the participants, even after the fact. But in ring signatures forward secrecy is recorded. We need to track these records if any private key is corrupted with the malicious

people. We can Use single server without key splitting and we use RSA instead of DSA with forward secrecy is explored in future work.

III IMPLEMENTATION

In our Architecture we are using secure key servers over cloud where the privacy of each key part is preserved by using Homomorphic algorithm. Here a user to log in with Facebook, linked in or other social network and take's his private key from a group of key servers. We are using Black Box, which is built for Crypto-Book framework. It allows a user to sign a file secretly, for example if they wish to leak a document, while offer proof of the document's sincerity in the form of a evidence that it comes from one of a significant group of Face book users**.** We are going to use DSA [12] and RSA [13] based on key pickup systems. DSA uses key division and RSA does not. For the RSA version we implemented a single key server and for the DSA version we implemented a three server group with the keys split over the three servers. Here the key servers have their private key encrypted part. If the key server compromises the secrecy of the key will not be known to the hacker**.**

3.1 Algorithm

We are using DSA based public/private keys. We are using Linkable Ring Signature scheme. We are assuming group G of order m. Then $A=G^B$ mod M. Where A is the public key and B is the private key. Digital signatures can be formed by the requester by getting required public and private keys from the key servers. This procedure will be done as follows. In this model we are assuming three key servers who are storing public/private keys. The user enters into face book by trusted third party or by providing OAuth token by the face book. If user using Mobile network app then facebook sends OAuth to that app. If the user authenticates successfully by the face book, the OAuth token will be sent to the key servers requesting for the user's private key. After private key request by the user key server makes request to face book to verify token in face books API. If verification succeeds then key servers checks its data base for the private key part of the desired user and sends it to him/her. In the similar fashion the public keys will be supplied to user except the OAuth token. The user can take any public key from key servers. Homomorphic algorithm is applied for each private key part by each client in order to provide more security for the compromised key servers. If the keys do already not exist with key servers with face book username, the key server generates them and stored in the data base. If the private key generated by the key server will be applied homomorphic algorithm by the user and that part is saved in the key server inorder to attain more security. For public key generated part directly stored into key server's data base. Whenever an initiator requests for the required public keys and his/her private key. When he/she gets the required public keys and his/her private key. He / She generates ring signature by using private/public key pairs and then sends this file through anonymous network

IV FUTURE WORK

Our future work is to use Autoregressive Moving Average Model (ARMA) model is to be applied for LRS for the previous and future prediction of attackers. Autoregressive model (AR) is a time-honored tool for tolerant and predict a time series data. It estimates the current term Z_k of the series by a linear weighted sum of previous *p* terms in the series. The model order *p* is generally much smaller than the length of the series. AR is often pooled with Moving-Average model (MA) to obtain compound ARMA model for usually better correctness. Cross-site correlation attack is going to threat the user's anonymity. This happens when third party sites may compromise. If a user from one group authenticates him/her to any third party sites, then the other groups may know about the user by violating the privacy of a user. Lot of groups are involved in the anonymous network there may be risk of privacy. Future work depends on the investigation of user privacy preservation. We can also use DSA with single key sever rather than key splitting in our future work. This is simple that entire key generated by key server and is encrypted by RSA algorithm is further encrypted

by the client using homomorphic algorithm and saved in the single key server. So if the key server compromises can not reveal the key to the attacker.

V CONCLUSIONS

Our model explains the security aspects of the private key of each user by splitting and applying homomorphic algorithm on each key part of a user and pacing each part on each key server. If more key splitting needs more key servers for a single user. Thus security increases but ambiguity increases. We cannot recover the composite key if any key server is busy with some other task. Autoregressive Moving Average Model (ARMA) model is to be applied for LRS for the previous and future prediction of attackers. Future work depends on the investigation of user privacy preservation. Our model concentrates more on security

REFERENCES

[1] E. Hammer-Lahav. The OAuth 1.0 protocol, Apr.2010. RFC 5849.

[2] OpenID. http://openid.net/.

[3] R. Rivest, A. Shamir, and Y. Tauman. How to leak a secret. In ASIACRYPT, pages 552{565,Dec. 2001.

[4] J. K. Liu, V. K. Wei, and D. S. Wong. Linkable spontaneous anonymous group signature for adhoc groups. In Australian Conference on Information Security and Privacy, pages 614{623, July 2004.

[5] D. I. Wolinsky, H. Corrigan-Gibbs, B. Ford, and A. Johnson. Scalable anonymous group communication in the anytrust model. In EuroSec, Apr. 2012.

[6] R. Dingledine, N. Mathewson, and P. Syverson. Tor: the second-generation onion router. In 12th USENIX Security, Aug. 2004

[7]Crypto-Book: An Architecture for Privacy Preserving. Online Identities. John Maheswaran

[8] J.-F. Raymond. Tra_c Analysis: Protocols, Attacks, Design Issues, and Open Problems. In Design Issues in Anonymity and Unobservability, July 2000.

[9] D. Kedogan, D. Agrawal, and S. Penz. Limits of anonymity in open environments. In 5th International Workshop on Information Hiding, Oct. 2002.

[10] G. Danezis and A. Serjantov. Statistical disclosure or intersection attacks on anonymity systems. In Information Hiding Workshop, May 2004. , David Isaac Wolinsky, Bryan Ford. Yale University,. {john.maheswaran

[11]Short Linkable Ring Signatures Revisited. Man Ho Au1, Sherman S.M. Chow2, Willy Susilo1, and Patrick P. Tsang3. 1 Center for Information Security Research.

[12] Federal Information Processing Standards Publication. Digital signature standard (DSS), July 2013. FIPS 186-4.

[13] R. L. Rivest, A. Shamir, and L. Adleman. A method for obtaining digital signatures and public-key cryptosystems. Communications of the ACM, 1978.

Authors signature

(O SRI NAGESH)