DATA CONFIDENTIALITY IN A CLIENT SERVER MODEL

Shubham Krishna Anshu\textsuperscript{1}, Sabreen Javeed\textsuperscript{2} and S.Kuzhalvaimozhi\textsuperscript{3}
\textsuperscript{1,2,3}Information science & Engineering, NIE College of Engineering

Abstract- The main objective of this paper is to ensure the confidentiality of data without using any secure channel. We have used the client server approach and implemented the same to get an idea of network based applications using the same concept. Much research has been conducted to preserve the privacy of data, we have tried the same and added a bit flavor to it. We first discuss the RMI concept followed by the encryption technique used and a secured way of storing the data at server side and lastly an efficient way of accessing this data at the client side.

Keywords: confidentiality, secure channel, RMI, encryption

I. INTRODUCTION

Confidentiality is the key model of privacy. It is a key concept of ensuring that data is not made available to unauthorized/unauthenticated people. It can be achieved by many cryptographic algorithms (encryption). It includes key agreement by client and server sides which is basically done through symmetric and asymmetric algorithmic approaches. Now we will discuss the key concepts used in the paper.

1.1. Remote method invocation

Although an Internet system provides a basic communication service, the protocol software cannot initiate control with, or accept contact from, a remote computer. Of course, two application involved in a communication cannot both wait for a message to arrive. One application must actively initiate interaction while the other application passively waits. Most network applications use a form of communication known as the client–server paradigm. A server application waits passively for contact, while a client application initiates communication actively. RMI (Remote Method Invocation) is a true distributed computing application interface for java, written to provide easy access to remote virtual machines. The concept uses serialization and remote procedure call to send information back and forth between remote objects.

![Figure 1.1. Client Server Model](image)

1.2. Cryptographic primitives

In addition to technological developments in artificial intelligence, multimedia processing and data interpretation, and to an easy and cheap access to the communication channel, the above services call for the adoption of security measures that ensure that the information provided by the users and the knowledge made available by the service providers are adequately protected.
To prevent the leakage of critical information the cryptographic primitives are used. Cryptography is basically about encrypting the plain text to get the cipher text using some sort of secret key and later to get back the plain text we decrypt the cipher text using the same secret key. In our application we have used the homomorphic encryption technique about which we will discuss later in the paper. The data at the server side can be kept in the encrypted format. Thus, avoiding the malicious attacks on the data. Further to access the data at the client side, the client need the secret key for decrypting. And only after the data is decrypted the client can read the data.

Since this is a client server approach so it allows multiple clients to access the data at the same time. But only after the data is decrypted by all the required clients, it is available for download and can be presented in the readable format. Things will get clear as we proceed.

1.3. Homomorphic encryption

Homomorphic Encryption (HE) allows direct addition and multiplication of cipher texts while preserving decrypt ability. That is, Enc (m1) Enc (m2) = Enc (m1 * m2), where Enc (m) stands for the cipher text of m, and plaintexts respectively. One could also try to solve our problem using this technique, but HE uses the same decryption key for original data and the aggregated data. That is, the operator who executes homomorphic operations upon the cipher texts are not authorized to achieve the final result. This forbids aggregator from decrypting the aggregated result, because if the aggregator is allowed to decrypt the final result, he can also decrypt the individual cipher text received, which contradicts our motivation. Also, because the size of the plaintext space is limited, the number of addition and multiplication operations executed upon cipher texts was limited until Gentry et al. proposed a fully homomorphic encryption scheme and implemented it in. However, Naehrig et al. pointed out in that the complexity of general HE is too high to use in real application. Naehrig et al. also proposed a HE scheme which

1.4. Actors

1.4.1. Admin

The admin will have control over the database of the application. The job of admin is to specify various users who are meant to receive the data in a secured way. User registration will be done by the admin and it will be stored in the database by their username and password. It is the job of admin to encrypt the data by using a secret key. The admin will specify the data to be sent to users and it will also specify the users for which this data is meant to be.

1.4.2. Users

The user can login to their individual account using the username and password assigned by the admin. The user will choose the option to decrypt the document. The document will be decrypted but it will not be available to the user to download since all the users for which the particular data was meant to be haven’t decrypted the same.

1.5. Implementation

The process starts with the admin who creates various departments and specifies various users in the department. The users are provided with their userid and passwords to access the application. Now, based on the need of various departments the admin will upload the data on the server side. The data is not stored in the server side as plain text instead it is stored as the cipher text. The encryption algorithm which we use here is Homomorphic encryption technique.

After the document is successfully uploaded and stored in the server. All the users from the respective department will get the notification regarding the data uploaded by the admin. The users will now decrypt the data by accessing the key from the server which is stored in the key file. It is only after all the members of that particular department have decrypted the data, any user can access the data in a readable format from the server.

In case some user is not available or out of laziness didn’t decrypt the data, in order to avoid any unfavorable consequences the data can be decrypted by the admin itself and later the information can be conveyed to the particular user.
II. ADVANTAGES

Since the system has admin as well as user authentication process, it is trustworthy and due to systematic operation, it is reliable in nature.

The application will be available on any java platform.

The application provides complete security for Security system through user credential since the comparison of the previously stored credential is done with the presently entered. Where the comparison is done based on stored credential and allows the user to access the application.

Since we are using the java platform to support our application maintenance is very easy and economical also.

Is the ability of making systems and organizations to work together (inter-operate).

III. DISADVANTAGES

The encryption technique used doesn’t provide verifiable computing and further it is not semantically correct.

The performance of the application suffers due to the use of homomorphic encryption because cipher texts in the ciphers you mention are much larger than the plaintexts, so communication requirements typically go up. The computations on these large cipher texts are typically slower than if you just performed the computation on the plaintext itself. Because of this, in the outsourcing computation model, we typically see a requirement that encrypting inputs and decrypting outputs should be faster than performing the computation itself. In the case of multiple parties with individual inputs this seems to be less of a concern as privacy, not efficiency is the concern.

Since the username and password pass through the communication channel as plain text it is very much prone to Replay attacks.

In the case of multiple, participating parties, guaranteeing fairness (which means everyone who is supposed to get an output, gets it) is often difficult and requires extra machinery (e.g., threshold decryption) and more assumptions (threshold of honest parties, etc). Another concern in the multiple, participating parties model is how the parties privately input their values for the computation. If one party knows the private key, they can decrypt the inputs and violate another party's privacy. So in this case, threshold decryption is often used. This in itself is a disadvantage as generating threshold keys is a non-trivial task.

IV. CONCLUSION

We have seen in this paper how we have proposed a network based application which not only discuss how client server model operates but also describes how we have ensured secure way of data flow through the channel. Further development can be done on the field by using means to avoid replay attacks such as flow of userid and password as message digests, using the combined concept of secure multiparty computation and homomorphic encryption to ensure data privacy.

REFERENCES


