Securely Sharing Encrypted Medical Information

Arnab Deb Gupta, Yuriy Polyakov, Kurt Rohloff, and Gerard Ryan
College of Computing Sciences
New Jersey Institute of Technology
Newark, NJ 07102
Email: {ad479, polyakov, rohloff, gwryan}@njit.edu

***Abstract*—Modern medical data information systems must collect and present data to authorized users. Distributed long-term medical data storage requires an effective security model for delegating data access. An effective delegation model must keep data encrypted at all times and avoid the need to share decryption keys. We present a secure information architecture for implementing such a model with end-to-end data encryption that restricts data access to designated recipients. Our architecture uses recent Proxy Re-Encryption (PRE) advances in a publish-subscribe design pattern that can be used to securely share medical data. The architecture would lower medical data storage costs by using commodity cloud hosting while reducing vulnerability to attacks that compromise confidentiality of sensitive medical records.**

I. Introduction

The contribution of this paper is discussion on a secure information architecture and prototype for the collection, processing, and distribution of encrypted medical data using a lattice-based variant of Proxy Re-Encryption (PRE). The architecture improves upon prior security models by providing end-to-end lattice encryption to encrypting data at its source, transmitting and storing the data on a cloud environment, and share data with recipients who can decrypt the data without having to share decryption keys or decrypt data in the cloud host.

II. Security Model Use Case

A use case for our PRE information architecture is a clinical patient that contributes their medical records to support medical research. The researcher's cloud data store provider needs to access the patient's records. Data cannot be stored in the clear. PRE enables data to be stored in an encrypted state at the cloud provider and the cloud host to convert stored patient data ciphertext into a new encryption that can only be decrypted with the emergency care provider's private key.

III. Lattice-Based PRE With PALISADE-PRE

Lattice-based PRE is effective for delegating access to medical data because the key under which data is encrypted can be switched without decrypting the data. Our architecture's public key PRE implementation is ”PALISADE-PRE”. It is based on the unidirectional PRE scheme. This type of encryption was originally defined in [1]. However, PALISADE-PRE provides a more recent approach to PRE based on lattice problems. Lattice encryption schemes such as ours are 1) secure against quantum computing attacks [2] and 2) asymptotically faster than other public key schemes such as RSA. We implemented PALISADE-PRE in C++ for its cross-platform support and allowances for low-level system optimizations and generic object oriented design. The library consists of four modules: the Encoding Layer supports representation of plaintext, the Crypto Layer provides APIs to use lattice based data constructs defined in the Lattice Layer, and lattice operations are decomposed primitive arithmetic operations on integers, vectors, and matrices in the Math Layer. The Math Layer also includes efficient algorithms and discrete Gaussian samplers.

IV. PRE Information Architecture Prototype

Our architecture's prototype is based on the medical data Use Case described in Section II and is a Java-based web implementation of the publish-subscribe model allowing 1) a Clinic to publish encrypted patient data to a PRE-enabled server and 2) an External Researcher to securely access and use data from that server through a subscription. The prototype uses PALISADE-PRE as its cryptosystem and uses JavaScript Object Notation (JSON) to stream and store ciphertext and keys used in cryptosystem operations.

Figure 1 illustrates the prototype's end-to-end PRE workflow. The PRE Server 1) creates a JMS Queue to publish ciphertext to Clinic Subscriber and 2) creates a JMS Queue to publish re-encrypted ciphertext to External Researcher 1 Subscriber. An MDEJB is deployed in each Subscriber to monitor their queue. The JMS Queue broker type allows asynchronous retrieval of published data. Each Subscriber decrypts the ciphertext published to them by the PRE Server. Encryption and decryption operations are performed with calls to the PALISADE-PRE library on each Client and re-encryption is performed with calls to the PALISADE-PRE library on the PRE Server. A Java servlet deployed in Clinic Publisher encrypts data by using local Java Native Interface (JNI) middleware to call PALISADE-PRE's encryption API. An MDEJB deployed in each Subscriber Client decrypts data by using local JNI middleware to call PALISADE-PRE's decryption APIs. The PRE Server's re-encryption function is triggered by a request (i.e. rPRE in Figure 1) from Clinic Subscriber to a Java Servlet on the PRE Server that uses local JNI middleware to call PALISADE-PRE's re-encryption API. The information architecture in Figure 1 can be extended to provide a multi-hop re-encryption scheme.

The prototype's key distribution scheme for PALISADE-PRE operations uses a public key infrastructure model. Each

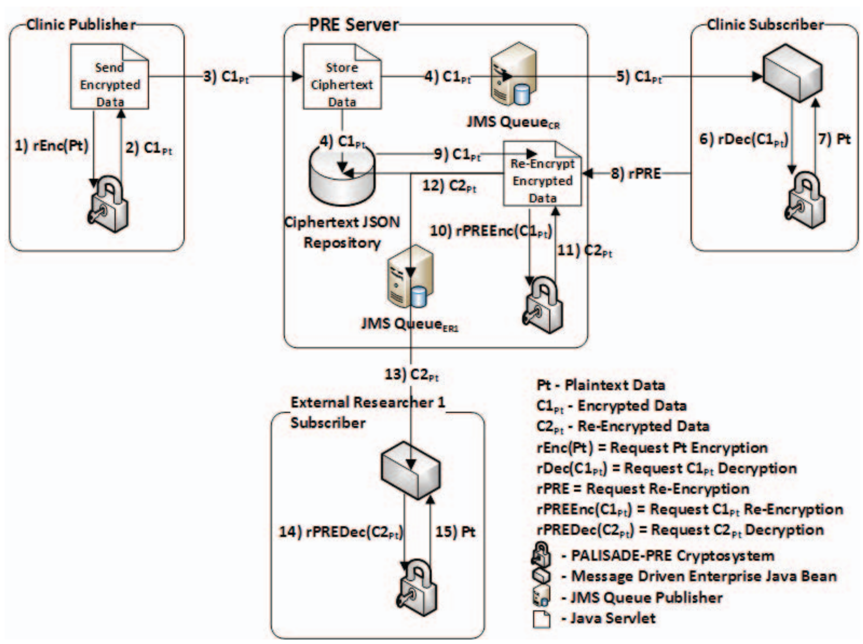

Fig. 1. Prototype PRE Information Architecture

Client generates a public/private key pair and the public key is sent to the scheme's Policy Authority in the PRE Server. The Clinic's public key encrypts patient medical data, the Clinic's private key decrypts encrypted patient medical data, and External Researcher 1's private key decrypts re-encrypted patient medical data generated by the PRE Server. It can be assumed that Clinic Subscriber has the Clinic's public/private key pair generated by Clinic Publisher. If External Researcher 1 needs access to encrypted patient data, they must first use their Client to generate their own public/private key pair. Clinic Subscriber can then generate the re-encryption key used by the PRE Server to re-encrypt ciphertext for External Researcher 1 by 1) retrieving External Researcher 1's public key from the Policy Authority and 2) calling PALISADE-PRE's PREKeyGen API with that public key and the Clinic's private key. Clinic Subscriber provides the re-encryption key to the PRE Server when it requests that re-encrypted ciphertext be generated for External Researcher 1. The PRE Server then calls PALISADE-PRE's ReEncrypt API to perform the re-encryption operation using the re-encryption key and the ciphertext stored for encrypted medical data to be shared with External Researcher 1. The PRE Server then publishes the re-encrypted ciphertext for External Researcher 1's JMS Queue. External Researcher 1 Subscriber would then use its private key to recover the shared patient medical data. Key generation operations are performed through a Servlet on each Client using JNI middleware to interface with PALISADE-PRE APIs for generating public, private, and re-encryption keys. This key distribution scheme can be extended to facilitate a multi-hop re-encryption scenario.

Our prototype satisfies the CIA triad's Confidentiality leg because 1) only encrypted data is transmitted between the PRE Server and its Clients and 2) it does not require keys to be shared for a party (e.g. an External Researcher) delegated data access to decrypt the data. The Confidentiality factor is enhanced by the use of quantum computing resistant lattice encryption techniques to compute the encrypted data transmitted between the PRE Server and its Clients [2]. A cryptographic signature scheme can be implemented over the components displayed in Figure 1 to satisfy the triad's Integrity leg. The public/private key pairs generated by prototype Clients provides the framework for the scheme and APIs for signature generation and verification must be added to PALISADE-PRE. The triad's Availability leg will be verified through activities: 1) stress, concurrency, and scalability testing, 2) guarding against server-related hardware issues and denial-of-service attacks through service replication strategies, and 3) guarding against data transmission losses or interruptions by implementing an effectively robust data backup and verification model.

V. Initial Experimentation

Experiments on a commodity laptop encrypted 1kb of data in 6.1ms, decrypted 1kb of data in 7.9ms, and performed access delegation operations for 1kb of data in 40ms. These results were obtained at a level of security with a work factor analogous to AES-128.

References

[1] G. Ateniese, K. Fu, M. Green, and S. Hohenberger, "Improved proxy re-encryption schemes with applications to secure distributed storage," *ACM Transactions on Information and System Security (TISSEC)*, vol. 9, no. 1, pp. 1–30, 2006.

[2] D. Micciancio, "Lattice-based cryptography," in *Encyclopedia of Cryptography and Security*. Springer, 2011, pp. 713–715.