Review of methods for secret sharing in cloud computing

Dnyaneshwar Supe **Amit Srivastav** **Dr. Rajesh S. Prasad**

Department of Computer Engineering
DCOER, Narhe, Pune-41
www.dcoerpune.org

Abstract:-

Cloud computing provides various IT services. Many companies especially those who are in their initial stages found this cloud IT service economically beneficial. This IT service paradigm still requires overcoming some security concerns before it can be fully deployed. This paper focuses on the various secret sharing algorithms that can be used in conjunction with cloud computing IT paradigm such as Public Key Cryptography, Ravest-Shimer-Adleman (RSA) Algorithm, Diffie-Hellman Algorithm, and then Elliptic Curve Cryptography (ECC). In this Paper, the author's show that using Public Key, Private Key and ECC how we can secretly share the data over the cloud environment. This paper also summarizes about which algorithm is best suited for cloud environment for secret sharing.

I. Introduction

In traditional data management models, people store and protect their own data under their own authority, whereas in cloud computing, the responsibility of data management and protection no longer belongs to the data owner. This fundamental change brings some new security challenges that traditional security solutions might not work.

One of the most difficult challenges is the protection of data privacy. Cloud's data center stores data from different clients. These data may physically reside in the same hardware.

Thus, cloud providers must have an efficient data isolation mechanism to prevent illegal data accesses from outsiders, other clients, or unauthorized cloud employees. The protection against malicious cloud employees is a difficult problem and may require a fully homo-morphic encryption algorithm.

Unfortunately, cryptologists were not able to find any such encryption algorithm for years. Actually, they are not even sure whether such encryption algorithm exists. Without homo-morphic encryption algorithms, it is unlikely for clients to have 100% data privacy against cloud employees.

II. Literature

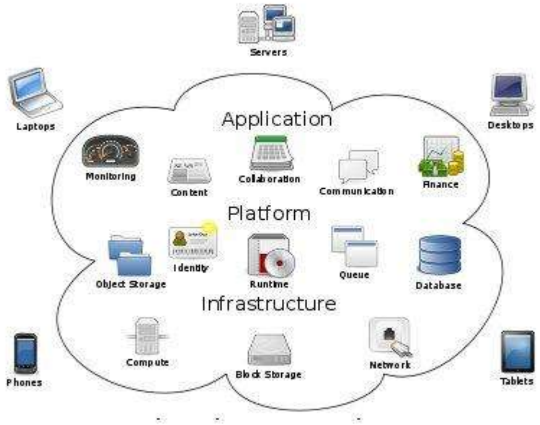


Figure 1: Cloud Computing

Cloud computing provides an environment where heterogeneous systems interacts over the internet. Heterogeneous systems involve different environments such as one system may use operating system such as windows, Linux, Macintosh etc.

Cloud computing is the use of computing resources (hardware and software) that are delivered as a service over a network (typically the Internet). The name comes from the use of a cloud-shaped symbol as an abstraction for the complex infrastructure it contains in system diagrams.

Cloud computing entrusts remote services with a user's data, software and computation. There are many types of public cloud computing [1]:

- Infrastructure as a service (**IaaS**)
- Platform as a service (**PaaS**)
- Software as a service (**SaaS**)
- Storage as a service (**STaaS**)
- Security as a service (**SECaaS**)
- Data as a service (**DaaS**)
- Business process as a service (**BPaaS**)
- Test environment as a service (**TEaaS**)
- Desktop as a service (**DaaS**)
- API as a service (**APIaaS**)

III. Issues on Data Protection

Regarding data protection, cloud computing raises a number of interesting issues. Data protection law is based on the premise that it is always clear where personal data is located, by whom it is processed and who is responsible for data processing.

Cloud computing appears to fundamentally conflict with this evidence. For example, if a customer uses an e-mail service based on cloud computing, the customer's data can be stored anywhere in the world, depending on where the servers on which the necessary storage capacity is available are located.

Different services supplied by a wide range of providers are regularly bundled to produce an end-user proposal, for example, if the mail service provider obtains the storage capacity required to store its customers' data from other providers. Therefore, with cloud computing it is no longer possible to say where the data is at a certain moment and by whom and how it is being processed.

This means that it is doubtful whether those responsible for data processing, in accordance with data-protection regulations, are in a position to effectively assume their responsibility at all.

If the data circulates freely around the globe via the internet, it is also no longer clear which data-protection authorities at which location are responsible for ensuring the observance of the principles of data protection.

The above discussion tells us about data protection issues in cloud computing. Now we need to encrypt the data for hiding or for privacy of the data for that different algorithms are used which are as follows [4].

1) Public Key Cryptography
2) RSA(Rivest-Shamir-adleman) or Shamir’s Secret Sharing Algorithm
3) Diffie-Hellman Algorithm
4) ECC(Elliptical Curve Cryptography) Algorithm

1. Public Key Cryptography

Public-key cryptography **[1]** refers to a cryptographic system requiring two separate keys, one of which is secret and one of which is public.

Although different, the two parts of the key pair are mathematically linked. One key locks or encrypts the plaintext, and the other unlocks or decrypts the ciphertext. Neither key can perform both functions. One of these keys is published or public, while the other is kept private.

There are two types namely Symmetric Public Key and Asymmetric public key.

1.1 Symmetric public key

The symmetric public key variations of which have been used for thousands of years - use a *single* secret key, which must be shared and kept private by both the sender and the receiver, for both encryption and decryption.

To use a symmetric encryption scheme, the sender and receiver must securely share a key in advance.

1.2 Asymmetric public key

Asymmetric key system in which neither party needs to even touch the other party's padlock (or private key)

A commutative cipher is one in which the order of encryption and decryption is interchangeable, just as the order of multiplication is interchangeable (i.e. A*B*C = A*C*B = C*B*A). A simple XOR with the individual keys is such a commutative cipher.

2. Rivest-Shamir-Adleman algorithm

RSA[1] is an Internet encryption and authentication system that uses an algorithm developed in 1977 by Ron Rivest, Adi Shamir, and Leonard Adleman.

The RSA algorithm is the most commonly used encryption and authentication algorithm and is included as part of the Web browsers from Microsoft and Netscape. It's also part of Lotus notes, Intuit's Quicken, and many other products.

2.1 How the RSA System Works

The mathematical details of the algorithm used in obtaining the public and private keys are available at the RSA Web site. Briefly, the algorithm involves multiplying two large prime numbers (a prime number is a number divisible only by that number and 1) and through additional operations deriving a set of two numbers that constitutes the public key and another set that is the private key.

Once the keys have been developed, the original prime numbers are no longer important and can be discarded. Both the public and the private keys are needed for encryption /decryption but only the owner of a private key ever needs to know it. Using the RSA system, the private key never needs to be sent across the Internet.

The private key is used to decrypt text that has been encrypted with the public key. Thus, if I send you a message, I can find out your public key (but not your private key) from a central administrator and encrypt a message to you using your public key.

When you receive it, you decrypt it with your private key. In addition to encrypting messages (which ensures privacy), you can authenticate yourself to me (so I know that it is really you who sent the message) by using your private key to encrypt a digital certificate.

When I receive it, I can use your public key to decrypt it. A table might help us remember this.

To do this	Use whose	Kind of key
Send an encrypted message	Use the receiver’s	Public key
Send an encrypted signature	Use the sender’s	Private key
Decrypt an encrypted message	Use the receiver’s	Private key
Decrypt an encrypted signature(and authenticate the sender)	Use the sender’s	Public key

3. Diffie–Hellman key exchange

Diffie–Hellman key exchange (D–H) is a specific method of exchanging cryptographic keys. It is one of the earliest practical examples of key exchange implemented within the field of cryptography.

The Diffie–Hellman key exchange method allows two parties that have no prior knowledge of each other to jointly establish a shared secret key over an insecure communication channel. This key can then be used to encrypt subsequent communications using a symmetric key cipher.

In 2002, Hellman suggested the algorithm be called Diffie–Hellman–Merkle key exchange in recognition of Ralf Markle's contribution to the invention of Public-Key Cryptography (Hellman, 2002).

4. Elliptic Curve Cryptography (ECC)

Elliptic Curve Cryptography [2] (ECC) is a public key cryptography. In public key cryptography each user or the device taking part in the communication generally have a pair of keys, a public key and a private key, and a set of operations associated with the keys to do the cryptographic operations. Only the particular user knows the private key whereas the public key is distributed to all users taking part in the communication.

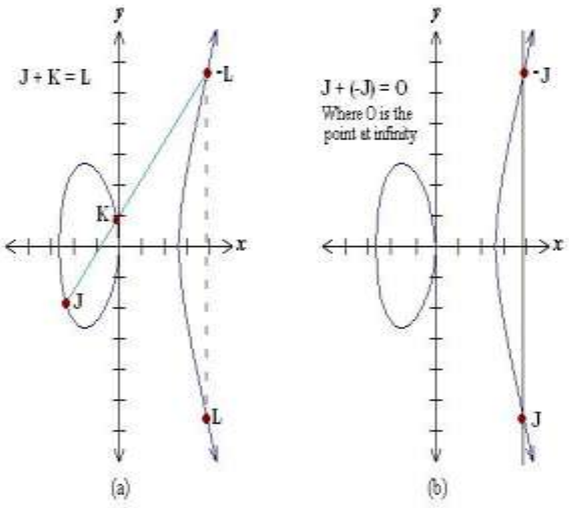
Some public key algorithm may require a set of predefined constants to be known by all the devices taking part in the communication. 'Domain parameters' in ECC is an example of such constants. Public key cryptography, unlike private key cryptography, does not require any shared secret between the communicating parties but it is much slower than the private key cryptography

. The mathematical operations of ECC is defined over the elliptic curve y2 = x3 + ax + b, where 4a3 + 27b2 ≠ 0. Each value of the 'a' and 'b' gives a different elliptic curve. All points (x, y) which satisfies the above equation plus a point at infinity lies on the elliptic curve.

The public key is a point in the curve and the private key is a random number. The public key is obtained by multiplying the private key with the generator point G in the curve. The generator point G, the curve parameters 'a' and 'b', together with few more constants constitutes the domain parameter of ECC.

One main advantage of ECC is its small key size. A 160-bit key in ECC is considered to be as secured as 1024-bit key in RSA.

4.1. Geometrical explanation



Consider two points J and K on an elliptic curve as shown in figure (a). If K ≠ -J then a line drawn through the points J and K will intersect the elliptic curve at exactly one more point –L. The reflection of the point –L with respect to x-axis gives the point L, which is the result of addition of points J and K. Thus on an elliptic curve L = J + K.

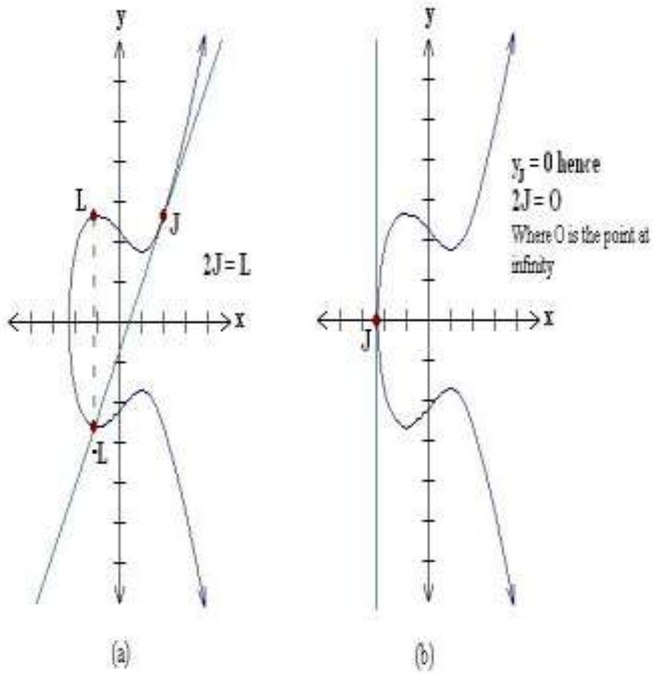
If K = -J the line through this point intersect at a point at infinity O. Hence J + (-J) = O. This is shown in figure (b). O is the additive identity of the elliptic curve group.

A negative of a point is the reflection of that point with respect to x-axis [2].

5. Point doubling

Point doubling is the addition of a point J on the elliptic curve to itself to obtain another point L on the same elliptic curve [2].

5.1. Geometrical explanation



To double a point J to get L, i.e. to find L = 2J, consider a point J on an elliptic curve as shown in figure (a). If y coordinate of the point J is not zero then the tangent line at J will intersect the elliptic curve at exactly one more point –L. The reflection of the point –L with respect to x-axis gives the point L, which is the result of doubling the point J.

Thus L = 2J. If y coordinate of the point J is zero then the tangent at this point intersects at a point at infinity O. Hence 2J = O when yJ = 0. This is shown in figure (b).

6. Finite Fields

The elliptic curve [2] operations defined above are on real numbers. Operations over the real numbers are slow and inaccurate due to round-off error. Cryptographic operations need to be faster and accurate. To make operations on elliptic curve accurate and more efficient, the curve cryptography is defined over two finite fields.

- Prime field F_p and
- Binary field $F_2{}^m$

The field is chosen with finitely large number of points suited for cryptographic operations. Section 7 and 8 explains the EC operations on finite fields. The operations in these sections are defined on affine coordinate system. Affine coordinate system is the normal coordinate system that we are familiar with in which each point in the coordinate system is represented by the vector (x, y) [2].

7. EC Cryptography

The EC algorithms are specified in SEC 1: Elliptic Curve Cryptography [2]. An overview of EC cryptographic algorithms for key agreement and digital signature are explained below [2].

7.1. ECDSA - Elliptic Curve Digital Signature Algorithm

Signature algorithm is used for authenticating a device or a message sent by the device. For example consider two devices A and B. To authenticate a message sent by A, the device A signs the message using its private key. The device A sends the message and the signature to the device B. This signature can be verified only by using the public key of device A. Since the device B knows A’s public key, it can verify whether the message is indeed send by A or not.

ECDSA is a variant of the Digital Signature Algorithm (DSA) that operates on elliptic curve groups. For sending a signed message from A to B, both have to agree up on Elliptic Curve domain parameters. The domain parameters are defined in section 8. Sender ‘A’ have a key pair consisting of a private key dA (a randomly selected integer less than n, where n is the order of the curve, an elliptic curve domain parameter) and a public key QA = dA * G (G is the generator point, an elliptic curve domain parameter). An overview of ECDSA process is defined below:

7.1.1 Signature Generation

For signing a message m by sender A, using A’s private key dA

1. Calculate e = HASH (m), where HASH is a cryptographic hash function, such as
SHA-1
2. Select a random integer k from [1,n − 1]
3. Calculate r = x1 (mod n), where (x1, y1) = k * G. If r = 0, go to step 2
4. Calculate s = k − 1(e + dAr)(mod n). If s = 0, go to step 2
5. The signature is the pair (r, s)

7.1.2 Signature Verification

For B to authenticate A's signature, B must have A’s public key QA

1. Verify that r and s are integers in [1,n − 1]. If not, the signature is invalid
2. Calculate e = HASH (m), where HASH is the same function used in the signature
generation
3. Calculate w = s −1 (mod n)
4. Calculate u1 = ew (mod n) and u2 = rw (mod n)
5. Calculate (x1, y1) = u1G + u2QA
6. The signature is valid if x1 = r(mod n), invalid otherwise [2]

8. ECDH – Elliptic Curve Diffie Hellman

ECDH [2] is a key agreement protocol that allows two parties to establish a shared secret key that can be used for private key algorithms. Both parties exchange some public information to each other. Using this public data and their own private data these parties calculates the shared secret. Any third party, who doesn’t have access to the private details of each device, will not be able to calculate the shared secret from the available public information. An overview of ECDH process is defined below.

For generating a shared secret between A and B using ECDH, both have to agree up on Elliptic Curve domain parameters. The domain parameters are defined in section 9. Both end have a key pair consisting of a private key d (a randomly selected integer less than n, where n is the order of the curve, an elliptic curve domain parameter) and a public key = d * G (G is the generator point, an elliptic curve domain parameter).

Let (dA, QA) be the private key - public key pair of A and (dB, QB) be the private key - public key pair of B.

1. The end A computes K = (xK, yK) = dA * QB
2. The end B computes L = (xL, yL) = dB * QA
3. Since dAQB = dAdBG = dBdAG = dBQA. Therefore K = L and hence xK = xL
4. Hence the shared secret is xK

Since it is practically impossible to find the private key dA or dB from the public key K or L, its not possible to obtain the shared secret for a third party.

IV. Proposed Security Solutions

The cloud computing is a virtual environment that requires transfer data throughout the cloud. Therefore, several data storage concern can arise. Typically, users will know neither the exact location of their data nor the other sources of the data collectively stored with theirs.

To preserve security of your cloud-based virtual infrastructure, perform security best practice at both the traditional IT and virtual cloud.

To ensure data confidentiality, authentication, integrity, and availability, the provider should include the following:

1. Encryption: the sensitivity of data may require that the network traffic to and from the virtual machine be encrypted, using encryption at the host OS software.

2. Physical security: keep the virtual system and cloud management hosts safe and secure behind carded doors, and environmentally safe.

3. Authentication and access control: the authentication capabilities within your virtual system should copy the way your other physical systems authenticate. One time password and biometrics should all be implemented in the same manner. Also authentication requires while you are sending data or message from one cloud to other cloud. To provide message authentication we will use digital signatures [2].

4. Separation of duties: as system get more complex, mis-configuration take place, because lack of expertise coupled with insufficient communication. Be sure to enforce least privileges with access controls and accountability.

5. Configuration, change control, and patch management: this is very important and sometimes overlooked in smaller organizations. Configuration, change control, patch management, and updated processes need to be maintained in the virtual world as well as physical world.

6. Intrusion detection and prevention: what's coming into and going out of your network has to know. A host based intrusion prevention system coupled with a hypervisor based solution could examine for virtual network traffic.

Among these proposed security solutions, we consider in this paper authentication and encryption for secure data transmission from one cloud to other cloud that requires secure and authenticated data with elliptic curve cryptography.

V. Conclusion

This paper presents various algorithms for secret sharing in cloud computing as well as how secret sharing happens using Elliptical Curve Cryptography algorithm in cloud computing.

In this paper we started with the cloud computing needs and what is cloud computing then we identified the algorithms that can be used for secret sharing in cloud computing. We explained each algorithm with an effective and easy to understand example including the role each algorithm plays in the secret sharing for cloud computing.

References

1) Jyh-haw Yeh" A PASS scheme in cloud computing " Dept. of Computer Science, Boise State University, Boise, Idaho 83725, USA
2) Anop ms""Elliptical Curve Cryptography" an implementation guide.
3) Wikipedia, the free encyclopedia
4) Dr Ursula Widmer-Dr Widmer & Partners "Cloud Computing and Data Protection",

Authors

DNYANESHWAR SUPE
Pursuing B.E. (Computer Engineering)
University of Pune, Dnyanganga College
of Engineering & Research, Survey
No.39, Narhe, Pune 411041, India.

AMIT SRIVASTAV
Pursuing B.E. (Computer Engineering)
University of Pune, Dnyanganga College
of Engineering & Research, Survey
No.39, Narhe, Pune 411041, India.

Dr. RAJESH PRASAD
Professor, ZES's Dnyanganga College of
Engineering & Research, Narhe, Pune