Abstract- Cloud computing is shared pools of configurable computer system through which people can share resources, services and information among the people often through the internet. Many security problems occur such as authentication, confidentiality and integrity. Data encryption is broadly used to protect the data over internet and ensure security. Different data encryption algorithms has been developed that the data transmitted via internet is secure and prevent from hackers or attackers. Both ECC and RSA algorithms describes a comparative analysis with example in this paper.

Keywords - Cryptography, Cloud Security, RSA, ECC, Elliptic Curve Digital Signature Algorithms (ECDSA).

I. INTRODUCTION

Cloud computing provides access to a multimedia of services through the web, with only a minimum of client resources needed in which smaller organizations have access to processing power, storage and business processes that were once only available to large enterprises. Rather than keeping files on a proprietary hard drive is managed by third parties access at remote locations. The cloud services include online business applications, online file storage, and social networking sites and on-demand network services allow to a shared pool of computing resources [1]. Due to these advantages every organization are transmitting their data to the cloud so that it requires for protecting data against unauthorized access, denial of services or alteration and so on. Cloud provides secure storage (databases hosted by the Cloud provider). Three important security goals of data include namely: Confidentiality, Availability and Integrity. In recent times, Cryptography is considered combination of three kinds of algorithms like 1.Symmetric-key algorithms, 2.Asymmetric-key algorithms, and 3.Hashing. In hashing algorithm, it ensures integrity of data with efficient manner [2].

In Data cryptography concept, it handles the content of the data were scrambled, such as audio, video, text and all that to create meaningless structure and unreadable form during transmission. The cryptography is used to protect the data from invaders. Both the asymmetric and symmetric algorithms recognize how to be used to encrypt data at cloud storage [3]. Cloud storage handles a bulky set of databases and for such a large database asymmetric-key algorithm’s performance is slower than compared to symmetric-key algorithms.

II.CLOUD SECURITY CHALLENGES [4]

Privilege user access: The Inherent level of problem presents even if the sensitive data proceeds outside the bounds of organization.

- Data location: The data storage location is not recognized to the user
- Data Segregation: In cloud the data lies in a shared environment with the data of the other customers. Chances of having malfunction are high during the course
- Recovery: The location of the data stored and segregation problems it becomes another challenge to recover the data rightfully to its owner.
- Investigation Support: This process becomes not possible to identify illegal activity in the cloud.

III.SECURITY SOLUTIONS

The security of the clouds infrastructure is performed to preserve into the crucial characteristics like authentication, availability and integrity. Therefore the service provider should follow certain ground written rules [5]:

- Encryption: To secure sensitive data, to protect under the breaches to secure against persistent threats to the data.
- Physical Security of the Cloud: The data centers are an attraction to the malicious users likewise a bank repository is an attraction of robbers. For this reason providing physical security have to be provided.
- Authentication and Access Control: Authentication is used by a client while the client needs to recognize that the server is system it claims to be.

M.Subhashini\textsuperscript{1}, Dr.P.Srivaramangai\textsuperscript{2}

\textsuperscript{1} Ph.D.Research scholar, Department of Computer Science, Marathu pandiyar College of Arts & Science, Thanjavur, Tamilnadu, India.

\textsuperscript{2} Associate Professor, Department of Computer Science, Marathu pandiyar College of Arts & Science , Thanjavur. Tamilnadu, India.

International Journal of Scientific Research in Computer Science Applications and Management Studies

IJSRCAMS

Volume 7, Issue 5 (September 2018)
IV. SECURITY ALGORITHMS

In general, Cryptography is the process of hiding information like system passwords, ATM cards, and secretes transferring data from one place to another are completed with cryptography. The main purpose of cryptography is maintaining data secure and prevent from unauthorized attackers or hackers. The reverse process of data encryption is decryption [7]. Encryption algorithms play a very important task to provide secure communication over the internet. Encryption is the fundamental tool for securing the data. The data converts into scrambled form using “the key” and only user has the key to decrypt the data. There are two techniques in Security algorithms like asymmetric Algorithm and Symmetric Algorithm. There are several security algorithms in the cryptography like Asymmetric algorithms- El Gamal, Diffie-Hellman, RSA, DSA, Symmetric algorithms - Blowfish, Homomorphic, Idea, DES, AES, RC5, and so on.

A. RSA

RSA is considered as a Public-Key cryptography algorithm [9]. This algorithm uses the product of two prime numbers. The prime factors should be kept secret, using RSA algorithm encrypt the data to make available security with the intention that only the concerned user can access it. To protect the data, it is not allowed unauthorized access to it. RSA Algorithm is an asymmetric public key algorithm it uses two different keys such as public key and private key, private key used for decryption process and public-key is used for encryption process. RSA algorithm provides encryption, Key generation and decryption. The security of RSA algorithm lies on integer factorization issue. A strongest key pair is generally used (A and B) to generate modules n. The condition of selection of A and B is both numbers. It provides complexity to factor n by using specific factoring technique (n=a*b), public key or encryption key (n,e), if can factor n it's simple to find out d. Otherwise the technique used for choosing prime number must be efficient and the main feature of RSA. The key size performs the strength of cryptosystem. RSA key size refers to the size of n. If a and b has larger size number with same length, it’s very hard to factor the product n. The key size is based on the security require even if the larger size it provides better security on algorithm. There are three steps of RSA Algorithm [8]:

- Key generation
- Encryption
- Decryption

Key Generation:

In key generation, two prime numbers consider (i.e.) p and q. It consists of a private key and a public key. The public key will be recognized to everyone. The value of n is evaluated and select a random encryption key e calculates the gcd and it must be equal to 1. After that, the decryption key d, the public key and private key have to be evaluated.

1. Generate two different primary keys P and Q.
2. Calculate the modules N=P*Q.
3. Calculate the \( \phi(n) = (p-1)(q-1) \).
4. Select the public exponent an integer e such that \( 1 < e < \phi(n) \) and \( \gcd(e, \phi(n)) = 1 \)
5. Calculate the private exponent a value of d such that \( d = e^{-1} \mod \phi(n) \)
6. Public key = [e,n]
7. Private key = [d,n]

Encryption process:

In encryption process, it signifies a plaintext in series of numbers modulo n. The encryption process to achieve ciphertext C from plaintext M is very simple. It is formulated as:

\[ C = M^e \mod n \]

Where C = ciphertext, M = message text, E = public key, D = private key. A symmetric FEK (File Encrypted Key) will be encrypted to transmit the file concurrently asymmetric public key will generated; both will be combined to form an encrypted FEK with a header file.

Decryption process:

The reverse process of encryption will be decryption. It can be generated using the formula:

\[ M = E^d \mod n \]

Where C = ciphertext, M = message text, E = public key, D = private key.

Example:

i) Prime Number P = 97, Q = 71.
ii) RSA Modules N= 97 x 71 = 6887
\[ \phi(n) = (97-1)(71-1) = 6720. \]
iii) Public key \( e = 2^4 \times 61 + 1 = 65537 \).
iv) Private key \( d \equiv e^{-1} \pmod{6720} \equiv 5633 \).
v) Message M = "PUBLIC KEY CRYPTOGRAPHY"
vi) Encryption \( E(M) \equiv M^{65537} \pmod{6887} \).
vii) Decryption \( D(M) = M^{65537} \pmod{6887} \).
viii) Benny sends Alex the message "TAMILNADU" as follows:
   
   - The Input text partition into segments of Size 1
     (the symbol `#` is used as separator).
     
     \[
     T = A \# M \# I \# L \# N \# A \# D \# U = 01010100 \#
     \ 01000001 \# 01001101 \# 01001001 \# 01001100 \#
     \ 01001110 \# 01000001 \# 01001001 \# 01010101
     \]
   
   x) Encryption into ciphertext \( c[i] = m[i]^e \pmod{N} \)
   
   - \( kG \) and \( kPb \) as cipher text to the receiver side.

A. ECC (Elliptic Curve Cryptography)

Elliptic Curve Cryptography (ECC) is a public key cryptography developed independently by Victor Miller and Neal Koblitz in the year 1985 [10]. In Elliptic Curve Cryptography we will be using the curve equation of the form

\[ y^2 = x^3 + ax + b \]

Since ECC is a public key cryptography, it requires a private key and a public key. The two communicating parties are considered Alex and Benny. Elliptic curve equation agrees upon a generator \( G \). Let Alex and Benny private keys be \( nA \) and \( nB \) in respective manner. Alex and Benny public keys are known by \( P_A = nAG \) and \( P_B = nBG \) in respective manner. Even if Alex transmits a message \( 'PM' \) to Benny, Alex utilizes Benny’s public key to encrypt the message. The cipher text is known by \( P_c = \{kG, P_{m} + kP_{b}\} \) where \( 'k' \) is a random integer for a same message the cipher text is produced different each time. Benny decrypts the message by subtracting the coordinate of \( 'kG' \) multiplied by \( nB \) from \( 'PM + kPb' \). \( P_m = \{P_m + kP_b - nBkG\}[11] \).

Encryption

1. Obtain the text to be send.
2. Its convert to corresponding ASCII values.
3. The ASCII value partition as [ASCII values, group size, group size, 1], this operation group the ASCII values with size known by group size with no overlapping and the sub lists that have size lesser than group size are left as it is without padding.
4. Each group obtained from the above step is converted into big integer values taking base as 65536. From Digits[Group of ASCII values, 65536]
5. Pad with 32 to the end of the list from the above step if the count of the above list is odd, to make it even for performing complete pairing. Each single pair will be an input to the ECC system as ‘PM’. We pad with 32 because 32 represent blank space in ASCII code.
6. Choose random \( k \) value, \( k = \text{Random value with range } 1 \text{ to } n-1 \). Calculate \( kG \) and \( kPb \) using Point multiplication operation.
7. Compute \( PM + kPb \) using point doubling or point addition as needed.
8. Transmit \( Pc = \{kG, PM + kPb\} \) as cipher text to the receiver side.

Decryption

1. The cipher text \( Pc \) is obtained.
2. The left part \( kG \) and right part of the \( Pc \) separately (\( PM + kPb \)).
3. Multiply with \( nB \) to the left part and subtract it from the right part to obtain \( PM \).
   
   \[ \{PM + kPb\} - nBkG = PM \text{ since } Pb = nB. \]

Subtraction operation can be converted to addition by multiplying with \(-1\) to the \( y \) coordinate. By applying point addition operation can be validated to obtain the mirror image point over the \( x \)-axis.
4. By forming set of ASCII values, the above operation will be provide the integer value and then convert it back to list of ASCII values. Integer Digits \([n, b]\) in Mathematics provides a set of the base \( b \) digits in the integer \( n \). Digits function and Integer Digits are opposite to each other so that the ASCII values are secured during encryption and decryption.
5. The list of ASCII values convert to its corresponding characters.

Example:

1. Curve Size: Small, Curve Type: Real number, Curve attributes: \( a=4 \), \( b=10 \), Curve: \( y^2 = x^3 + 4x + 10 \). Point \( P = (0.55|3.51) \), Point \( Q = (0.45|3.44) \), Point \( R = P + Q = (4850.03|337767.03) \).
2. Curve Size: Large, Curve Type: \( F(p) \), Select curve attributes: ANSI X9.62.Curve: prime192v1, Radix: 16 hexadecimal, Curve attributes: \( y^2 = x^3 + 4x + 10 \), where
   
   \( a = \text{fffffffffffffffffffffffffffffffeffffffff}, \)
   \( b = 64210519e59c80e70fa7e9ab72243049feb8deecc146b9b1 \),
   \( P = \text{fffffffffffffffffffffffffffffff} \).

IJSRCSAMS

Volume 7, Issue 5 (September 2018) www.ijsrcsams.com
order of G : ffffffff15194e9079f99def836146bc9b1b4d22831
Base point G: Point P
x = 9011da03d515aba05057145e2ab54a75f002408d176438
y = 60882d7e4dc4d376f3b5486a5baba7ef07e2c774dca0023d
Base point G: point Q
x = 274acd0cf69376da44bd378515764aed90fda8e408ca86ca
y = d4b7c2640fcf20f0cf026d5817ffedc6229e2b9964d4eb4
Point R : R = P + Q
x= e676896cefc37b007e6e77586207bb9253df576360d8162f
y= 506b2782b796feca4f2cddf3a9a94de954fcd8b720f38b

3. Curve Size: Large, Curve Type: F(2^m), Select curve attributes : ANSI X9.62, Curve: c2pnb163v1, Radix : 16 hexadecimal
a = 72546b5435234a2e0789675f432ec89435de5242
b = c9517d06d5240d3cff38c74b20b6cd4d6fd9
m = 163
Base Point P:
 x = 0000006880337df7400fc26 b58941d5 f279f388
 f470743d
y = 00000059324f03b bbfa8a7b 0b32aa5e ffb6ed60
 691a6d7b
Base point Q:
X= 00000021c3be282 3ae1f1c5c c28a605c a0f07557
e26a006d
Y= 000000261250191 fceeb558 701993d9 2b2dc418
35dea27c
Point R : R = P + Q
X= 000000013e39a17 f0ab0115 a9b0d582 c3765040
4aad370c
Y= 00000002f6607c40 99d08bf5 67c2ff68 4475ae53
c15e2c33

4. Curve Size: Small, Curve Type: F(2^m), curve attributes : m=4, f = x^4+x+1,a=1,b=1,Curve: y^2 + xy = x^3 + x^2 + 1 ,
Point P = (g9|g8), Point Q = (g10|g5), Point R = P + Q = (g12|g4)

C. ECDH – Elliptic Curve Diffie Hellman

In ECDH key agreement protocol, it enables two communicating parties to establish the connection exchange some public information to each other using a shared authentication key [12]. Using this public and private data these parties computes the shared authentication key. But third party does not have access to the private details of each device, will not be able to calculate the shared authentication from the available public information. Using ECDH, shared secret is authenticated between A and B to agree up on Elliptic Curve domain parameters. 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).

Key Agreement Algorithm[13]

For establishing shared secret between two device A and B
Step1. If dA and dB are the private key of device A and B respectively. Private keys are random number less than n, where n is a domain parameter.
Step2. If QA = dA*G and QB = dB*G are the public key of device A and B respectively, G is a domain parameter
Step3. A and B exchange their public keys
Step4. The end A computes K = (xK, yK) = dA*QB
Step5. The end B computes L = (xL, yL) = dB*QA
Step6. Since K=L, shared secret is chosen as xK

Example:
Step 1: Set public parameters
Curve type: F (p), Curve Size: Small, Domain parameters:
\(a=4, b=10, p=13\), generator \(G=(3,6)\)

Step 2: Choose Secrets

\(\text{Alex}=2, \text{Benny}=3\)

Step 3: Generate shared keys

Secret key (d): \(Q=d*G\), \(\text{Alex}=(6,9), \text{Benny}=(5,5)\)

Step 4: Exchange shared keys

Step 5: Generate common key

\(\text{Key}=sA*QB\) and \(\text{key}=sB*QA\)

\(S=(6,4)\)

Elliptic Curve Digital Signature Algorithms (ECDSA) [14]

In Signature algorithm, the device authenticates and verifies a message or a signature. Two devices (A and B) authenticate a message transmit by A using its private key. The device A transmits the signature and the message to the device B. The device B recognizes A’s public key, it can verify whether the message is indeed transmitted by A or not. Elliptic curve groups transmit a signed message from A to B and then have to agree up on Elliptic Curve domain parameters. Transmitter A contain 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) [15]. An overview of ECDSA process is described below:

**Signature Generation**

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

1. Calculate \(e = \text{HASH} (m)\), where \(\text{HASH}\) 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 \((r, s)\)

**Signature Verification**

For B to authenticate A’s signature, B should 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 \(e = \text{HASH} (m)\), where \(\text{HASH}\) 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]

**Example:**

**ECDSA Key Generation**

Signature originator: subhashini subhashini

Domain parameters to be used ‘EC-prime239v1’:

Chosen signature algorithm: ECSP-DSA with hash function SHA-1

Size of message \(M\) to be signed: 9 bytes

Bit length of \(c + \) bit length of \(d = 477\) bits

Message \(m = "TAMILNADU"\)

Elliptic curve \(E\) described through the curve equation: \(y^2 = x^3 + ax + b \mod p\):

\(a = 88342353238919216479164875036030888531447659725\)

\(296036279245086069999836\)

\(b = 73852521740699241734859608803878172416486097179\)

\(709897189124042363193866\)

Private key = 1521284887

Public key \(W=(Wx,Wy)\) (W is a point on the elliptic curve) of the signature originator:

\(Wx=1224700832616806964197507145004472595353783052\)

\(77085576222307449754331974\)

\(Wy=13684505523169076698312319020790037819123632\)

\(304735280089313480758778\)

Calculate a 'hash value' \(f\) (message representative) from message \(M\), using the chosen hash function SHA-1.

\(f = 9818597392590116665238978628668698485340036048\)

**ECDSA Signature:**

\(G\) has the prime order \(r\) and the cofactor \(k\) (\(r*k\) is the number of points on \(E\)):

\(k = 1\)

Point \(G\) on curve \(E\) (described through its (x,y) coordinates):

\(Gx=1102820037495488548763485335341162577905061\)

\(504888124240194511594420911\)

\(Gy=86907840743550937874735187379305886850210384\)

\(946040694651368759217025454\)

\(r=88342353238919216479164875036030888480755034169\)

\(162775275345424702807307\)

The secret key \(s\) is the solution of the EC discrete log problem \(W=x^sG(x, y)\) (unknown)

\(S=70780634036932777419100170634676837891836172354\)

\(7822444392804388830078\)

Signature:

\(c=79198570642783917906561923343897820085157291\)

\(132345834893851920800314287\)

\(d=3102068971432915577478513154990659336945441085\)

\(9379176919291821100655367\)

IJSRCSAMS

Volume 7, Issue 5 (September 2018) www.ijsrscsams.com
**ECDSA Verification:**

If c or d does not fall within the interval [1, r-1] then the signature is invalid:
c and d fall within the required interval [1, r-1].

Calculate the number h = d^(-1) mod r:
h = 548569369109120940227468225601421950228370195378080992283112954075601388

Calculate the number h1 = f*h mod r:
h1 = 1224699657295685629530859

Calculate the number h2 = c*h mod r:
h2 = 4860385047375938925218996

Calculate the elliptic curve point P = h1 G + h2 W (If P = (Px, Py) = (inf, inf) then the signature is invalid):
P = 7919857063442783917906561923334389782008515729

Convert the group element Px (x co-ordinates of point P on elliptic curve) to the number i:
i = 7919857063442783917906561923334389782008515729

Calculate the number c’ = i mod r:
c’ = 7919857063442783917906561923334389782008515729

If c’ = c then the signature is correct; otherwise the signature is invalid: Verify results by comparing the two numbers c’ and c.

**V. EXPERIMENTAL RESULT**

Compare cryptography algorithms RSA, ECC, ECDH and ECDSA using a 4GB RAM and Intel core i5-2450 M CPU 2.50GHz machine. For encryption and decryption operations 5542 KB block size is used. By applying test data the security algorithms is evaluated in terms of the execution time required to store or retrieve the text data at cloud. The Simulation program inputs are: Algorithm and data block. Subsequent to a successful execution i.e. encryption and decryption process generate an efficient result. The analytical table is formed after the successful encryption / decryption process. To make sure that all the data are processed in the precise way. Basically, it is depend upon execution time (Encryption and Decryption time) as parameters. Given below Table 1 provides the key generation, as well as encryption and decryption times for RSA and ECC, ECDH, and ECDSA.

![Graph](image1.png)

**Figure 1: Key generation time**

**Encryption and Decryption Time**

Figure 2 and 3 shown, Encryption and decryption performance for the various algorithms are difficult to measure and are heavily influenced by system architecture and software/hardware optimizations. Compared to RSA, ECC offers better key pair generation performance, RSA requiring several time to generate large primes when compared to much smaller ECC key pair, RSA encryption lesser than ECC at RSA bit lengths of 1024 and above, although ECC decryption several times faster than RSA, finally both are efficient enough not to grant a system traffic jam problems. The ECDH and ECDSA algorithms provides similar processing time as ECC because both algorithm implementation is similar to ECC, but both take longer due to multiple exchanges steps are involved. Based on the result
shown in figure 2 and 3 the ECDSA algorithm better in both encryption and decryption process compare to other algorithms.

![Figure 2: Encryption time](image1)

![Figure 3: Decryption time](image2)

### VI. CONCLUSION

Cloud computing is the main challenge to solve data security issue. Encryption is the most excellent security technique, different types of encryption technique apply in cloud computing environment, some extend hacking can be prevented. RSA and ECC, ECDSA, ECDH security algorithms are compared study on in cloud using cryptographic techniques. Different algorithms utilize different security techniques other than they all are liable to different situations. The single security algorithms cannot be trusted one. Some of cryptography algorithms need each level in cloud applications.

### REFERENCES


[5]. Mr.MilindMathur and Mr. AyushKesarwani “Comparison between DES, 3DES, RC2, RC6, Blowfish and AES” Proceedings of National Conference on New Horizons in IT - NCNHT 2013


[7]. Anop ms”Elliptical Curve Cryptography” an implementation guide.


