ACQUIRING DIGITAL SIGNATURES FROM ELLIPTIC CURVE CRYPTOGRAPHY

REVIEW III

FINAL REPORT

Submitted by

Siddharth Chatterjee - 19BCE2249
Ishan Sagar Jogalekar - 19BCE2250
Arkaprava Mahato - 19BCT0173
Yash Sharma - 19BCT0195
Sulayam Abdul Rehman - 19BCE2474

Prepared For

INFORMATION SECURITY ANALYSIS AND AUDIT (CSE 3501)

PROJECT [J] COMPONENT

Submitted To

DR. RAJA S. P

Associate Professor Level Ao1

School of Computer Science and Engineering

Vellore Institute of Technology
(Deemed to be University under section 3 of UGC Act, 1956)

Table of Contents

1. AIM

The aim of our project is to construct a program which can calculate and verify the electronic digital signature based on the Elliptic Curve Cryptography.

SHA-1 will be used to calculate the hash function.Develop the ECDSA algorithm and implement it in Java Programming language

The elliptic curve cryptosystems are paid more and more attention because its key string is shorter and its security is better than other public cryptosystems.

The digital signature system based on elliptic curve (ECDSA) is one of the mainstream digital signature systems.

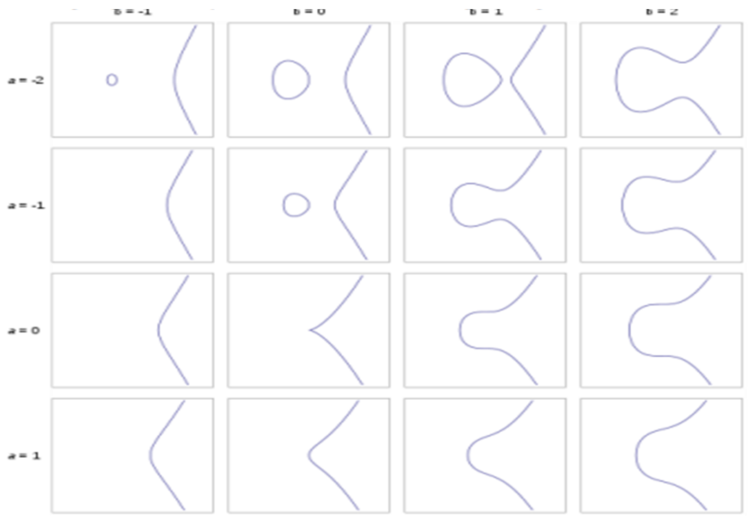
Elliptic Curve Digital signature represents one of the most widely used security technologies for ensuring un-forge-ability and non-repudiation of digital data. Its performance heavily depends on an operation called point multiplication.

Furthermore, the root cause of security breakdown of ECDSA is that it shares three points of the elliptic curve public ally which makes it feasible for an adversary to gauge the private key of the signer.

2. OBJECTIVE

The steps involved in ECDSA are formation of key-pair, signature-generation and signature-verification. The digital signature is typically created using the hash function. The transmitter sends the encrypted data along with a signature to the receiver. The receiver in possession of the sender's public key and domain parameters can authenticate the signature.

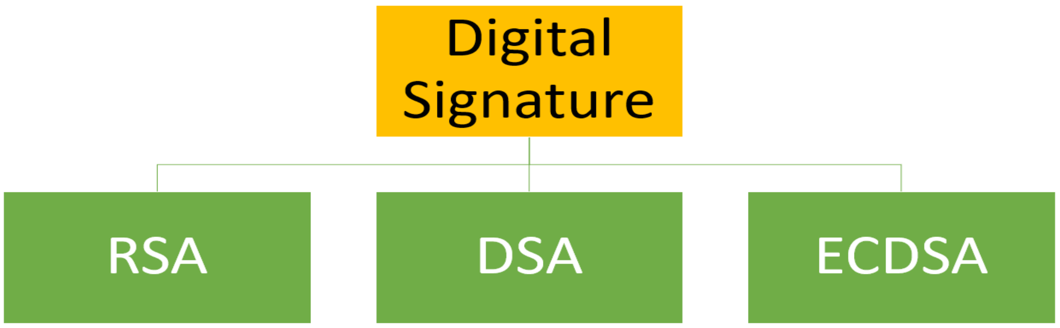
This is
The prime q of the finite field Fq, the equation of the elliptic curve E, the point P on the curve and its order n, are the public domain parameters. Furthermore, a randomly selected integer d from the interval [1, n-1] forms a private key. Multiplying P by the private key d, which is called scalar multiplication, will generate the corresponding public key Q.
The pair (Q, d) forms the ECC public-private key pair with Q is the public key and d is the private key. The generating point G, the curve parameters ‘a’ and ‘b’, together with few more constants constitute the domain parameters of ECC.
The public key is a point on the curve and the private key is a random number selected by the signer. The public key is obtained by multiplying the private key with the generating point on the curve.



Base Equation of curve - $y^2=x^3+ax+b$

3. METHODOLOGY

Digital signature is the method of securing documents and providing authentication , non-repudiation and message / document integrity.
It is implemented using asymmetric cryptography techniques, where 2 keys are generated.
When Sender uses its Private key (PR_A) to encrypt a message or document then this process is known as Digital signature method.
There are two main algorithms used in the current scenario 1.RSA and 2.DSA and we implemented ECC with basics of DSA, performed as ECDSA.



Encryption in this methodology is performed using elliptical curve cryptography(ECC) and the signing of messages is performed on the basis of the DSA algorithm. The key exchange algorithm is done using the Diffe-Helman key exchange algorithm.

DSA algorithm -

1. Key Generation:
    - Choose 'q' prime no also known as Prime Divisor.
    - Another prime no 'p' choose such that (p-1) mod q = 0.
    - Choose integer g ($1<g<p$) with condition,
        1. $g^p \bmod p = 1$
        2. $g = h^{((p-1)/q)} \bmod p$
    - h - Hash digest and H - Hash function
    - X is a private key - a random integer such that $0<X<q$.

- Y is public key $Y = g^X \bmod p$
- Private key package {p,q,g,X}
- Public key package {p,q,g,Y}

2. Signing generation:
   - Choose random no. k , $0<k<q$
   - $r = (g^K \bmod p) \bmod q$
   - $s = [k^{-1} (H(m)+XR)] \bmod q$
   - Signature Package {r,s}.
   - It will be sent as {M,r,s}, M is message and signature will append with message.

Verification done using verification component value V and value r.

<u>Elliptic Curve Operations</u>-

A. ***Prime Field*** :

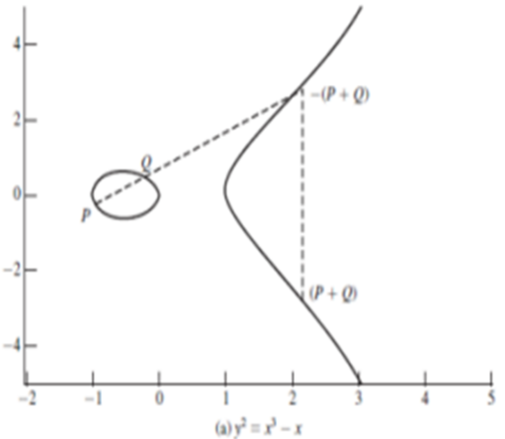
The equation of the elliptic curve on a prime field Fq is V2(modq)=x3+a×x+b where 4a3+27b2(mod q)≠0. Here the elements of finite field are integer between 0 and q−1. All operations such as point-addition, point-subtraction, point-division and point multiplication involve integers between 0 and q−1. The prime q is chosen such that there is finitely large number of points on the elliptic curve to make the cryptosystem secure.

B. ***Binary Field***:

The equation of the elliptic curve on a binary field Fm2 is y2+x×y=x3+ax2+b, where b≠0. Here elements of a finite field are integers. These elements are chosen such that length of each should be at most m bits. These numbers can be regarded as a binary polynomial having degree m-1. In binary polynomials the coefficients can only be o or 1. All operations involve polynomials of degree m-1 or lesser. The m is chosen such that there is a finitely large number of points on the elliptic curve to make the cryptosystem more secure.

• *Point Addition*:

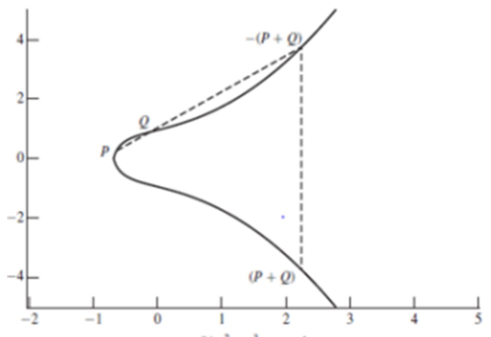
It is possible to obtain a third point R on the curve given two points P and Q with the aid of a set of rules. Such a possibility is termed as elliptic curve point addition. The symbol represents the elliptic curve addition P3=P1+P2. Point addition should not be confused with scalar addition.



(a) $y^2 = x^3 - x$

• *Point Multiplication*:

Consider a point P(xp,yp) on elliptic curve E. To determine 2P, P is doubled. This should be an affine point on EC. Equation of the tangent at point P is: S=[(3x2p+a)/2yp](mod p). Then 2P has affine coordinates (xr,yr) given by: xr=(S2−2xp) mod p yr=[S(xp−xr)−yp](mod p). Now 3Pcan be determined by point addition of points P and 2P, treating 2P=Q. P has coordinates (xp,yp) and Q=2P has coordinates (xq,yq). Now the slope is: S=[(yq−yp)/(xq−xp)] mod p P+Q=−R xr=(S2−xp−xq) mod p yr=(S(xp−xr)−yp) mod p. Thus k×p can be calculated by a series of point-doubling and point-addition operations.



(b) $y^2 = x^3 + x + 1$

For implementation of ECDSA we have used the P-192 Curve of ECC algorithm,where all base parameters are already defined.

A prime field is the field GF(p), which contains a prime number p of elements. The elements of this field are the integers modulo p; the field arithmetic is implemented in terms of the arithmetic of integers modulo p. The applicable elliptic curve has the form $y^2 = x^3 + ax + b$.

A binary field is the field GF(2m), which contains 2m elements for some m (called the degree of the field). The elements of this field are the bit strings of length m; the field arithmetic is implemented in terms of operations on the bits. The applicable elliptic curve has the form $y^2 + xy = x^3 + ax^2 + b$.

Although there is a virtually unlimited number of possible curves that meet the equation, only a small number of curves is relevant for ECC. These curves are referenced as NIST Recommended Elliptic Curves in FIPS publication 186. Each curve is defined by its name and domain parameters set, which consists of the Prime Modulus p, the Prime Order n, the Coefficient a, the Coefficient b, and the x and y coordinates of the Base Point G(x,y) on the curve.

4. LITERATURE REVIEW

LITERATURE REVIEW 1

Implementation of Elliptic Curve Digital Signature Algorithm
International Journal of Computer Applications (0975 – 8887)
Volume 2 – No.2, May 2010
Authors : Aqeel Khalique || Kuldip Singh || Sandeep Sood
Department of Electronics & Computer Engineering, Indian Institute of Technology Roorkee

LITERATURE REVIEW 2

RSA Encryption and Digital Signature Algorithm

2018 4th International Conference on Science and Technology (ICST), Yogyakarta, Indonesia

Authors : Farah Jihan Aufa || Endroyono || Achmad Affandi Department of Electrical Engineering Institut Teknologi Sepuluh Nopember Surabaya, Indonesia

LITERATURE REVIEW 3

An Efficient Elliptic Curve Digital Signature Algorithm (ECDSA)

2013 International Conference on Machine Intelligence Research and Advancement

Authors : Shweta Lamba || Monika Sharma

Technological Institute of Textile & Sciences Bhiwani, India

5. STEPS AND ALGORITHM

ECDSA Algorithm-

The equation of the curve of ECC is in cubic form , $y^2 = x^3 + ax + b$, P- 192 curve with 192 bits is one of the best basic curves for development of digital signature using ECC.

Common parameters in algorithm:

G - Curve base point , generate subgroup of large prime order of n

n - n is a random prime number. satisfies n x G = O O is an identity element.

G{x,y} coordinate on.

H is a hash function applied on message M to get a hash digest h.

Hash function can be used as SHA family or MD or even whirlpool algorithms.

For more security purposes SHA-256 is one of the best techniques for hash digestion.Here SHA-1 is used to avoid complexity.

- **Key generation:**
  1. Choose a random big prime integer under the curve (inside of curve) that is the private key PR_A.
  2. Public key , $PU_A = PR_A$ x G - G is the base point for the curve in terms of x,y axis coordinates. $PU_A = Q_A$
- **ECDSA signing:**
  1. Input msg + PRA of sender will be appended to get signature or to generate signature.
  2. Add hash function on message and calculate its hash digest h.
  3. Random no. k should satisfy condition $1<k<(n-1)$ where n can be considered as the endpoint prime number of the curve.
  4. Random point P calculated as $P = k * G$
  5. Now select r as r = P.X that is only X coordinate value of P as r in Digital signature.Represent X coordinate of P as R.
  6. Calculate signature proof: $s = k^{-1}(h(m) + PR_A * R) \bmod P$.
  7. k^{-1} is modulo multiplicative inverse of k and $(k^{-1}*k) \bmod p = 1$.
  8. Now the signature is appended with a message as {M,r,s}.

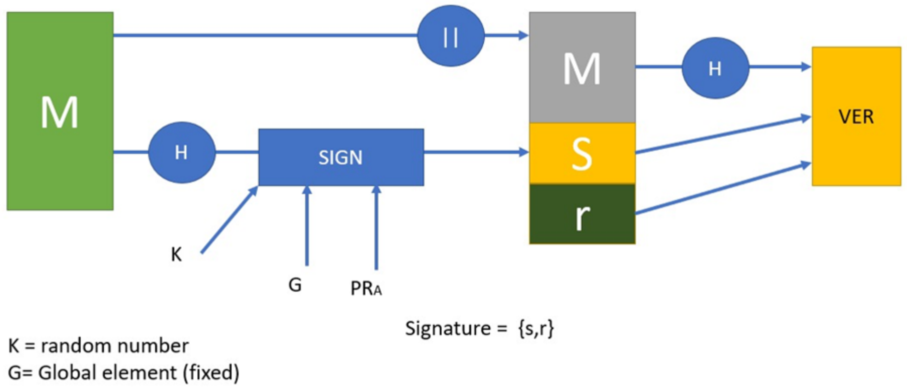
- **Verifying signature:**
    1. Verified signature is to check message integrity and checking whether a message is damaged or attacked or not during transmission.
    2. Calculate hash of message received using the same Hash function.
    3. Received message is like {M,r,s}.
    4. $s1 = s^{-1} \pmod n$
    5. $R` = (h*s1)*G + (r*s1) * PU_A$ PU_A - Public key of sender.
    6. $r` = R`.X$ it is X coordinate of R` only.
    7. now check for $r` = r$ if matched then signature is verified.

Another approach for verifying signature is also very feasible and oriented with curve equation -

Received message as {M,r,s}

Public keys and parameters of the curve are already shared with the receiver.

1. Calculate point P on the curve using curve parameter and public key.
2. $P = s^{-1}*z*G + s^{-1} * r * PUA$
3. z is a hash digest on message H(M) = h.
4. If X coordinate of matches with r received then signature is verified and message is integrated and safe.



Algorithm for ECDSA -Overview :

The transmitter sends the encrypted data along with a signature to the receiver. The

receiver in possession of sender's public key and domain parameters can authenticate

the signature. The prime q of the finite field Fq, the equation of the elliptic curve E, the point P on the curve and its order n, are the public domain parameters.

Furthermore, a randomly selected integer d from the interval [1, n-1] forms a private key. Multiplying P by the private key d, which is called scalar multiplication, will generate the corresponding public key Q.

The pair (Q, d) forms the ECC public-private key pair with Q is the public key and d is the private key. The generating point G, the curve parameters ‘a’ and ‘b’, together with few more constants constitute the domain parameters of ECC.

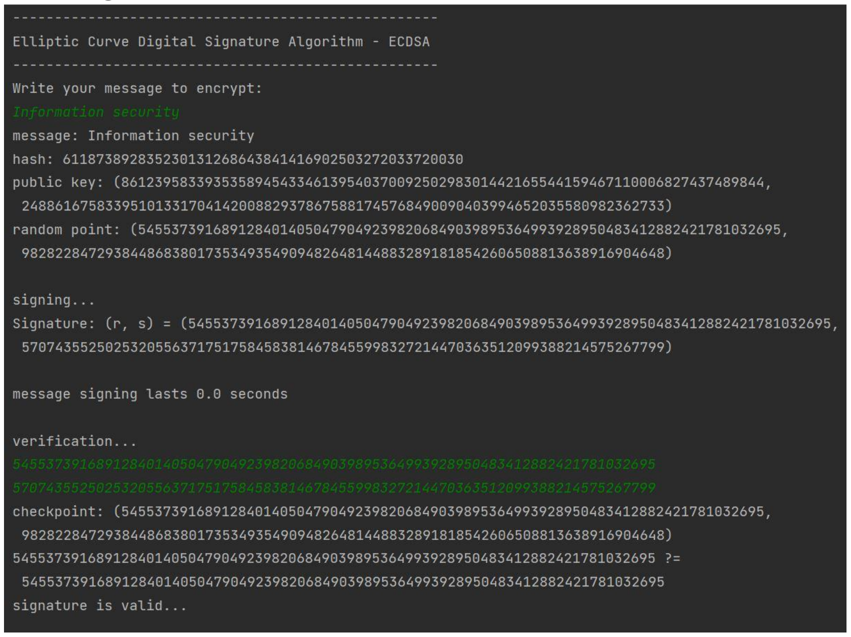
The public key is a point on the curve and the private key is a random number selected by the signer. The public key is obtained by multiplying the private key with the generating point on the curve.

6. CONCLUSION

The whole system is developed using Java programming language which provides better object oriented implementation and some inbuilt math functions about power and modulus.
The main purpose of developing this cryptographical based algorithm and implementation is to create a secure digital signature algorithm and provide a better solution over the existing algorithm (RSA and DSA).

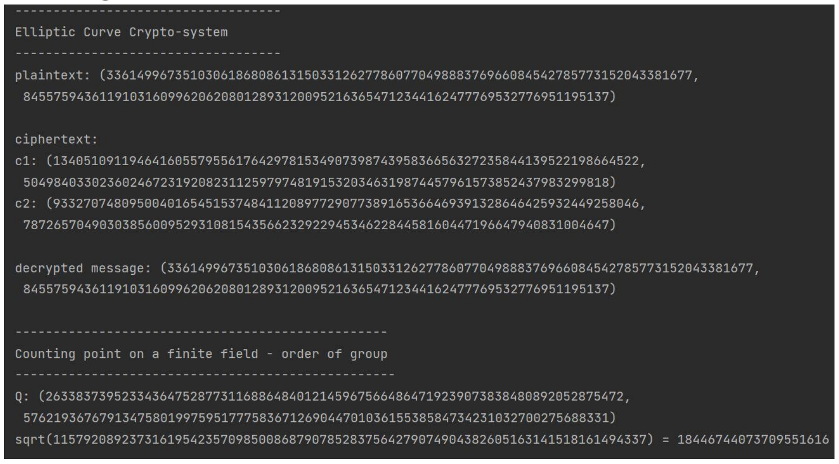
Outcomes from implementation:

1. Verification of signature-



```
---------------------------------------------------
Elliptic Curve Digital Signature Algorithm - ECDSA
---------------------------------------------------
Write your message to encrypt:
Information security
message: Information security
hash: 611873892835230131268643841416902503272033720030
public key: (86123958339353589454334613954037009250298301442165544159467110006827437489844,
 24886167583395101331704142008829378675881745768490090403994652035580982362733)
random point: (54553739168912840140504790492398206849039895364993928950483412882421781032695,
 98282284729384486838017353493549094826481448832891818542606508813638916904648)

signing...
Signature: (r, s) = (54553739168912840140504790492398206849039895364993928950483412882421781032695,
 57074355250253205563717517584583814678455998327214470363512099388214575267799)

message signing lasts 0.0 seconds

verification...
54553739168912840140504790492398206849039895364993928950483412882421781032695
57074355250253205563717517584583814678455998327214470363512099388214575267799
checkpoint: (54553739168912840140504790492398206849039895364993928950483412882421781032695,
 98282284729384486838017353493549094826481448832891818542606508813638916904648)
54553739168912840140504790492398206849039895364993928950483412882421781032695 ?=
 54553739168912840140504790492398206849039895364993928950483412882421781032695
signature is valid...
```

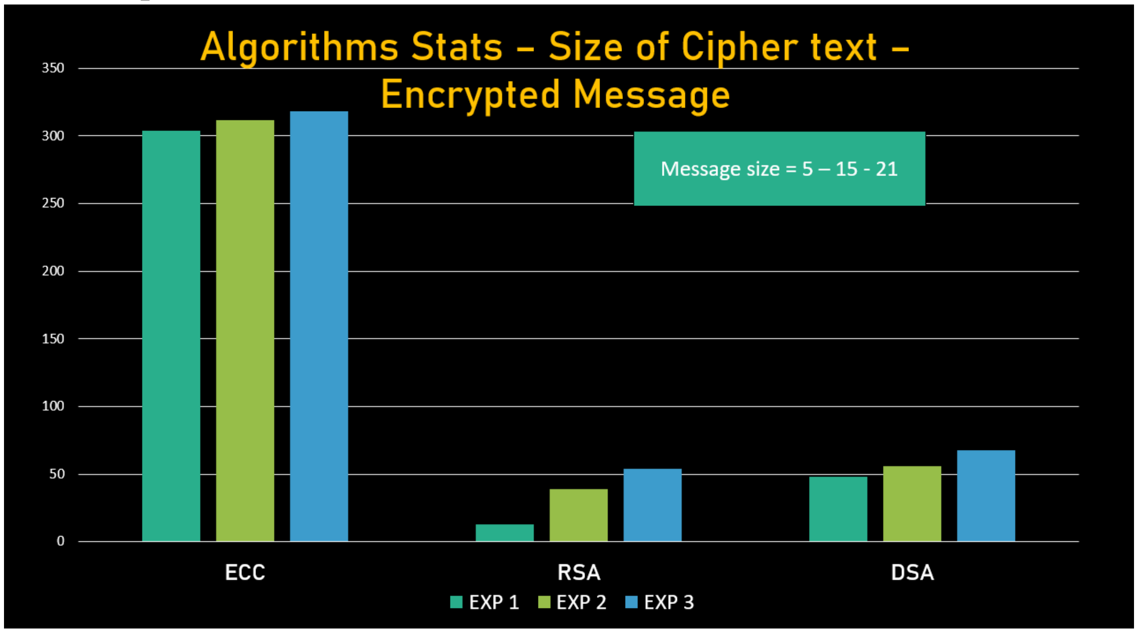
2. Encrypted message-



```
-----------------------------------
Elliptic Curve Crypto-system
-----------------------------------
plaintext: (33614996735103061868086131503312627786077049888376966084542785773152043381677,
 84557594361191031609962062080128931200952163654712344162477769532776951195137)

ciphertext:
c1: (13405109119464160557955617642978153490739874395836656327235844139522198664522,
 50498403302360246723192082311259797481915320346319874457961573852437983299818)
c2: (93327074809500401654515374841120897729077389165366469391328646425932449258046,
 78726570490303856009529310815435662329229453462284458160447196647940831004647)

decrypted message: (33614996735103061868086131503312627786077049888376966084542785773152043381677,
 84557594361191031609962062080128931200952163654712344162477769532776951195137)

-------------------------------------------------
Counting point on a finite field - order of group
-------------------------------------------------
Q: (2633837395233436475287731168864840121459675664864719239073838480892052875472,
 57621936767913475801997595177758367126904470103615538584734231032700275688331)
sqrt(115792089237316195423570985008687907852837564279074904382605163141518161494337) = 18446744073709551616
```

Graphs:

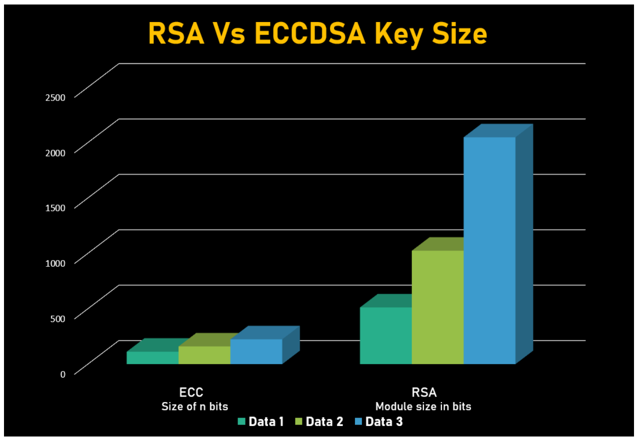
1. Size of cipher text -



The cipher text or encrypted message text length of ECDSA algorithm is much larger than RSA and DSA. This is due to the choosing of arbitrary large points on the ECC curve which is symmetrical to the X-axis but always increasing along with the Y-axis.

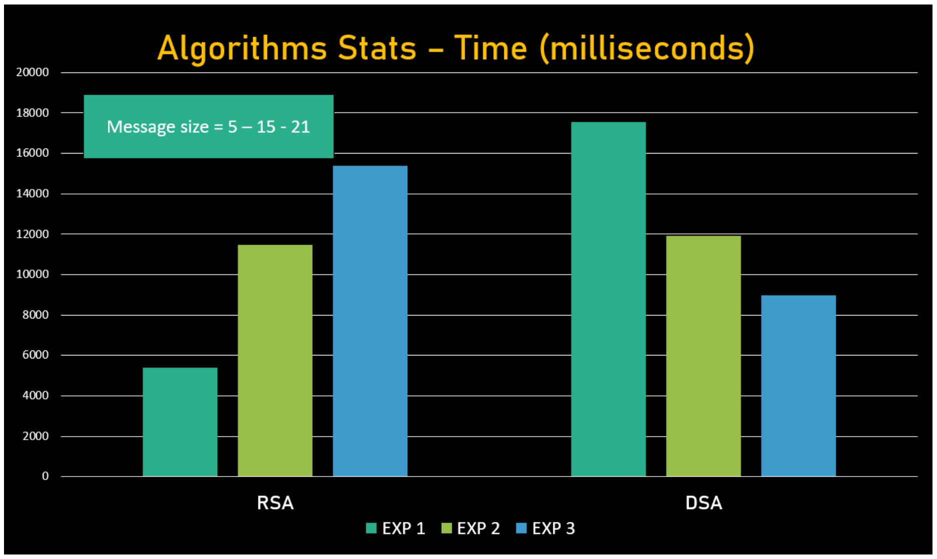
The more the length of the ciphertext provides more security against attacks because it takes time to decrypt it.(not in ethical way)

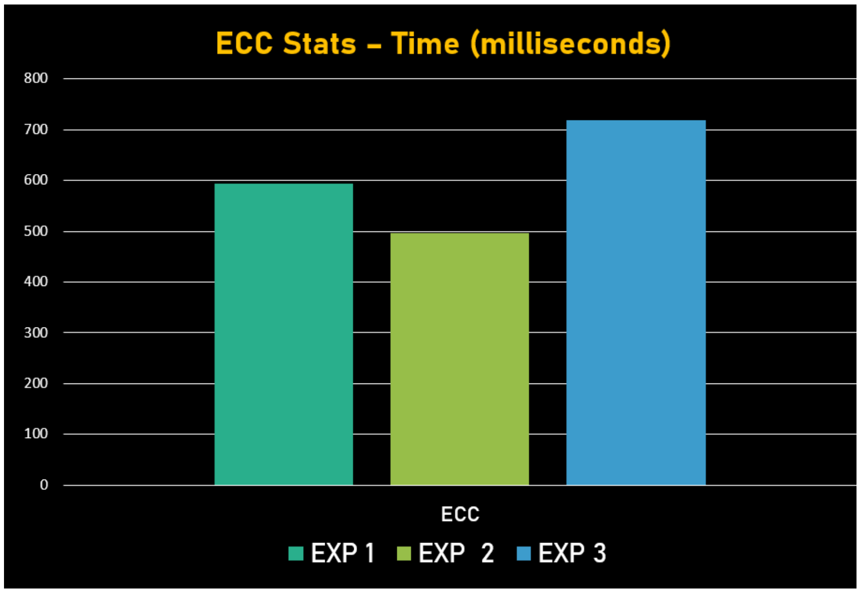
2. Size of the key -



The key size in ECDSA is way less than RSA but also provides the same security as RSA with less key size.The ECDSA signature string is 2 times longer than its private key.

3. Time taken -





These graphs clearly explain that the time taken during ECDSA algorithm for encryption and signing verification is less than for RSA and DSA.

ECDSA over other algorithms:

- Less time is more secure.
- Size of the cipher text is more.
- Creates two points as mentioned in the algorithm so, needed 2 functional points to validate signature.
- Key size is also less than RSA , gives more security

ECDSA security:

- Difficult in brute force than RSA because random points selected are also discrete log generated.
- It is difficult to brute force on large no of values even on RSA but RSA is a bit older technique and also inbuilt packages available.
- DSA encryption brute force is difficult but not impossible if string length is limited and repetitive then try-error method can be applied.
- Birthday attack ECC - might think a 64-bit hash is secure
- but by Birthday Paradox is not
- The Birthday Attack exploits the birthday paradox –
- The chance that in a group of people two will share the same birthday – only 23 people are needed for a Pr>0.5 of this.

  - In this case the Birthday paradox is not possible because of the 2 points.

7. FURTHER ADVANCEMENTS

- Currently, the implementation is done using Java programming language so it is very useful to include this type of digital signature while developing Web applications for handling secure sharing of files.

- Providing some changes in implementation will be useful to create Java package , JAR(Java Archive) libraries to include such encryption mechanisms with other implementations.

- In Case of mobile development and other security packages like spring security and maven, gradle dependencies we can also integrate this.This can be useful in encrypting and providing password lock system for PDF files while sharing over different social medias (like whatsapp,email etc.)

- As per for development this is implemented as ECDSA for digital signature but it can be developed for asymmetric encryption also.

- The statistical study is done over RSA, DSA and ECDSA which provides better cases for implementation algorithms.Also, helpful to understand important features of ECDSA.

8. REFERENCES

1: Alese, B.K., 2004. Design of Public key cryptosystem using elliptic curve. Ph.D. Thesis. Federal University of Technology Akure, Akure, Nigeria.

2: ANSI X9.62, 1999. Public key cryptography for the financial services industry. The Elliptic Curve Digital Signature Algorithm.

3: Arazi, B., 1993. Integrating a key distribution procedure into the digital signature standard. Electr. Lett., 29: 966-967.
Direct Link |

4: Aydos, M., 2000. Efficient wireless security protocols based on elliptic curve cryptography. Ph.D. Thesis. Oregon State University Ohio, USA.

5: Beguin, P. and J. Quisquater, 1995. Secure acceleration of DSS signatures using insecure server. LNCS, 917: 249-259.

6: Beth, T. and F. Schaefer, 1991. NonSuper Singular Elliptic Curves for Public Key Cryptosystems. Springer Verlag, Berlin, pp: 316-327

7: Diffie, W. and M.E. Hellman, 1976. New directions in cryptography. IEEE Trans. Inform. Theory, 22: 644-654.
CrossRef | Direct Link |

8: Diffie, W. and M. Hellman, 1976. Multiuser cryptographic techniques. Proceedings of the AFIPS National Computer Conference, June 7-10, 1976, ACM, New York, USA., pp: 109-112

9: El-Gamal, T., 1985. A Public Key Cryptosystem and a Signature Scheme Based on Discrete Logarithms. In: Advances in Cryptology, Blakley, G.R. and D. Chaum (Eds.). Springer-Verlag, Berlin, Heidelberg, pp: 10-18

10: IEEE STD 1363-2000, 2000. Standard specifications for public-key cryptography. Http://grouper.ieee.org/groups/1363/index.html.

11: ISO/IEC 14888-3, 1998. Information technology-security techniques-digital signatures with appendix-Part 3. Certificate Based-Mechanisms.

12: Johnson, D. and A. Menezes, 1999. The elliptic curve digital signature algorithm (EDSA). Technical Report CORR 99-34.

13: Kahn, D., 1967. The Codebreakers. Macmillan Co., New York

14: Koblitz, N., 1987. Elliptic curve cryptosystems. Math. Comput., 48: 203-209.
Direct Link |

15: Kravitz, D.W., 1993. Digital signature algorithm. U.S. Patent # 5,231,668, 27 Jul.