EdDSA OVER GALOIS FIELD GF (P^m) FOR IMAGE AND VIDEO FRAMES

Shivani Y N[1], Srinivas A[1], Thanmayi B K[1], Vignesh V[1], Dr. Srividya B V[2]
[1] Student, [2] Associate Professor,
Department of Telecommunication Engineering,
Dayananda Sagar College of Engineering, Bengaluru, Karnataka, India.

Abstract: The Edwards-curve Digital Signature Algorithm (EdDSA) was proposed to perform fast public-key digital signatures and thus replace the Elliptic-Curve Digital Signature Algorithm. Its key advantages over the latter include higher performance and straightforward, secure implementation for embedded devices. EdDSA algorithm is implemented over Galois Field. The operations like addition and multiplication in Galois field are different compared to normal addition and multiplication. Hence, implementing EdDSA over Galois field provides more security compared to the conventional EdDSA signature.

Index Terms – EdDSA, ECDSA, Galois field, Authentication.

I. INTRODUCTION

The digital signature is a digital code which is computed and authenticated by a public key encryption (like DSA, ECDSA) and is then attached to an electronically transmitted document for verification of its contents and also the identity of the sender [1].
The Edwards-curve Digital Signature Algorithm is a variant of Schnorr's signature system with Edwards's curves that may possibly be twisted [2]. The public-key signature algorithm EdDSA is similar to ECDSA which was proposed by Bernstein. EdDSA is defined for two twisted Edwards's curves which are edwards25519 and edwards448 [3]. EdDSA needs to be instantiated with certain parameters. Creation of signature is deterministic in EdDSA and it has higher security due to intractability of some discrete logarithm problems. Thus, it is safer than DSA and also ECDSA which require high quality randomness for each and every signature computed [8].

Generally, a point P= (x, y) lies on E, a twisted Edwards curve if it verifies the following formula: $ax^2 + y^2 = 1 + dx^2y^2$ where a, d are two distinct, non-zero elements of the field M over which E is defined. It is untwisted in the special case where a=1, because the curve reduces to an ordinary Edwards curve. Consider 'a' and 'd' values in the above equation as 10 and 6 respectively. The equation becomes $\rightarrow 10x^2+y^2=1+6x^2y^2$[4]. For which the plot of Edward curves is shown below:

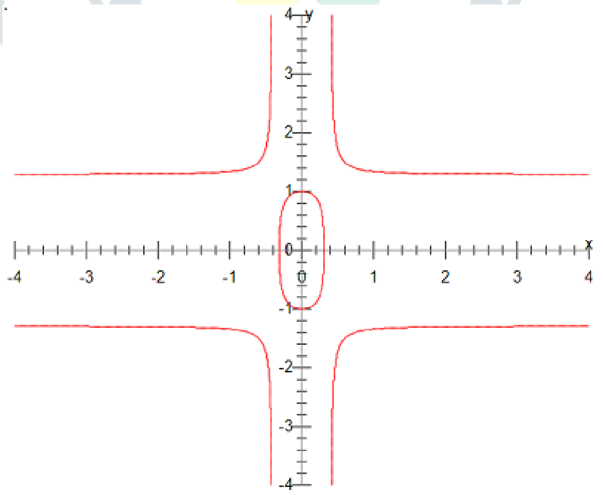

figure 1. plot of Edward curves for a=10 and d=6

Galois Field, named after Evariste Galois, also known as finite field, refers to a field in which there exist a finite number of elements [5]. It is very useful in translating computer data so that they are represented in binary forms. That is, computer data which is limited to a combination of two numbers, 0 and 1are the components in Galois field with number of elements being two. By representing data as a vector in a Galois Field, we can easily perform scrambled mathematical operations [6]. The elements of Galois Field GF (p^m) can be defined as:

GF(p^m) = (0,1,2,...,p−1) ∪(p,p+1,p+2,...,p+p−1) ∪(p^2,p^2+1,p^2+2,...,p^2+p−1) ∪...............∪(p^{m-1},p^{m-1}+1, p^{m-1}+2,...,p^{m-1}+p−1)

Where p∈P and m∈Z^+. The order of the field is given by p^mwhile p is called the characteristic of the field. GF refers to Galois Field. Also, the degree of polynomial of each element is at most m−1. For example GF(4) = (0, 1, 2, 3) which consists of 4elements in which each of them is a polynomial of degree '0' (a constant) while GF(2^4) = (0, 1, 2, 3.....*15*)and consists of 2^4= 16 elements where each of them is a polynomial of degree at most 2. GF (2^4) defines the basic arithmetic operations over the finite set of bytes.

II. METHODOLOGY

2.1 Key Pair Generation Process

For the EdDSA authenticator to function, it needs to know its own private key. The public key is obtained from the private key and the parameters specified over domain. The private key is not accessible to any third party. The public key must be openly read accessible. The public and private key pair ensures data is protected during transmission. A random integer is used to generate the private key which is known only to sender. The random number is passed to EdDSA algorithm which computes public key.

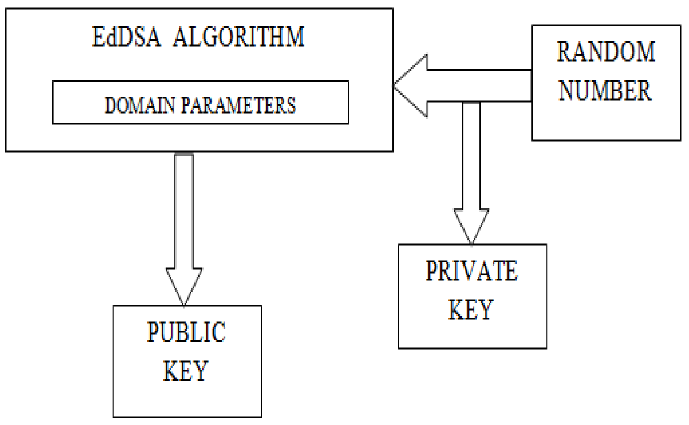

Figure 2.1 Public and Private Key generation

2.2 Signature Computation Process

The digital signature allows the receiver of a message to verify the message's authenticity using the sender's public key. It also offers non-repudiation, that is, the source of the message cannot deny the validity of the data sent. The message digests calculated from SHA Algorithm [9] are of fixed length and along with private key are used by parameters of EdDSA algorithm to compute the digital signature.

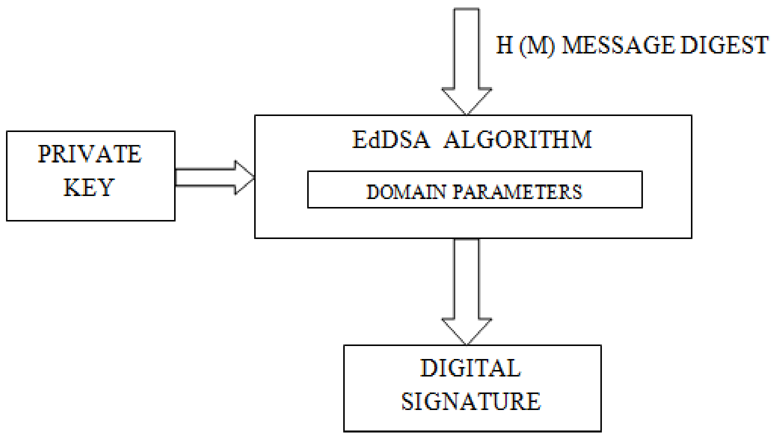

Figure 2.2 Generation of digital signature

2.3 Signature Verification Process

The signature verification is the counterpart of the signature computation. Its purpose is to verify the message's authenticity using the authenticator's public key. In order to reduce frauds, signature verification is very important. It increases accuracy and efficiency. This is performed at the message receiver end. The algorithm requires public key, received versions of digital signature and message digest.

Output determines the validity of the signature based on correct key given as input. If receiver enters matching key, then signature is verified. Else, signature verification fails.

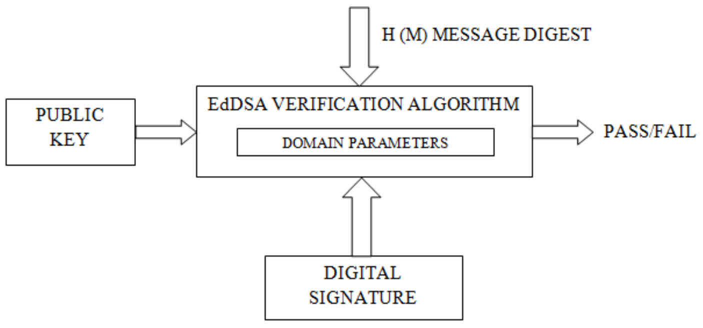

Figure 2.3 Verification of digital signature

2.4 Flow diagram for signing and verification

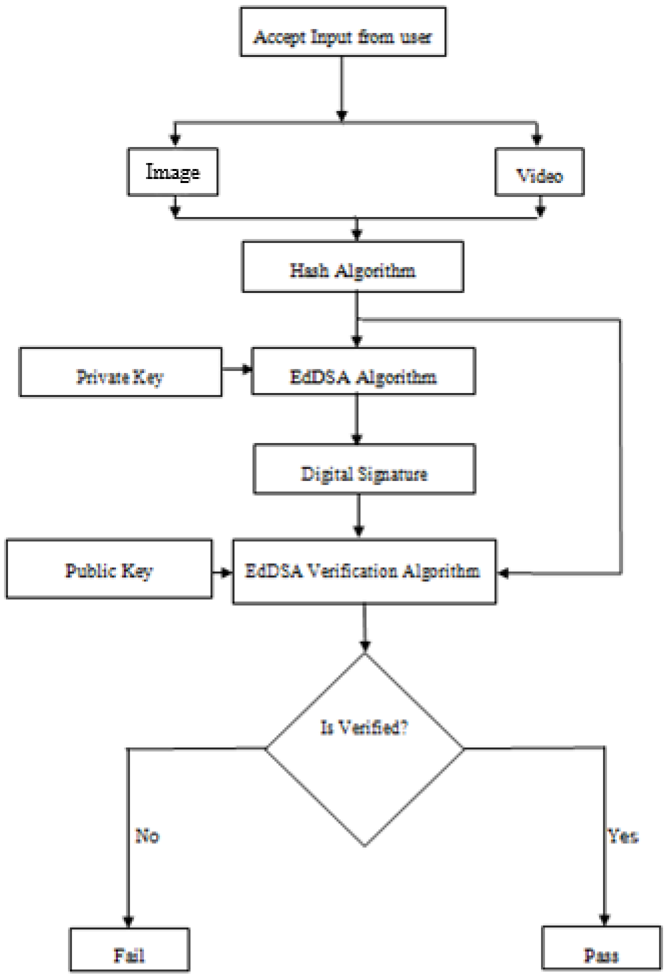

Figure 2.4 Flow chart for signing and verification

2.5 Edwards-Curve Digital Signature Algorithm

EdDSA is a public-key signature algorithm similar to ECDSA proposed by Bernstein et al [7]. In the paper RFC 8032, EdDSA has been defined for two twisted Edwards curves edwards25519 and edwards448; but, the EdDSA can also be instantiated over other curves.

In practice the public key and the signatures are output according to the encoding defined in [3]. Since there is a one-to-one relation between curve elements and encoded values we do not detail the encoding in our description. The signature (R, S) of a message M is computed according to following algorithm [4].

2.5.1 EdDSA signature algorithm

Requires: Message (M), hash digest values of message → (h_0, h_1,………, h_{2b-1}), Private key (B) and Public key (A) derived from B

```
1: a ← $2^{b-2} + \sum_{3 \le i \le b-3} 2^i h_i$
2: h ← H ($h_b$,....., $h_{2b-1}$, M)
3: r ← h mod GF($l$)
4: R ← r · B
5: h ← H (R, A, M)
6: S ← (r + ah) mod GF($l$)
7: return (R, S)
```

EdDSA uses a private key that is b-bit long and a hash function H that produces a 2b - bit output. One common instance is to use SHA-512 for b = 256 bits. Current popular hashes produce hash values of length n = 128 (MD4 and MD5) and n = 160 (SHA-1) [14].

An n-bit hash is a map from arbitrary length messages to n-bit hash values. An n- bit cryptographic hash is an n-bit hash which is one-way and collision-resistant. Such functions are important cryptographic primitives used for such things as digital signatures and password protection.

In this paper, Hash algorithm used is SHA-1. Length of Hash, of 2b bit length is 20 bits. So, private key is of 10 bits.

An integer ‘a’ is determined from H(k) = (h_0,h_1,...,h_{2b-1}) with “a = $2^{b-2} + \sum_{3 \le i \le b-3} 2^i h_i$”.The public key A is then computed from the base point B ≠ (0 , 1) of order l, chosen as per the EdDSA specifications [2], such that A = a·B. Where l can be any abstract algebraic equation, for example, $l = a^2 + b^2$,which is defined over Galois field p^m. ‘p’ is a positive prime number raised to value ‘m’.

For this algorithm, using Galois field for the calculation of value ***‘l’*** gives more security and reduces computational time.

Even if multiple signatures are computed for an identical message, same signature is obtained. Thus, EdDSA is deterministic in nature.

2.5.2 Verification of EdDSA signature

A signature is considered valid if it satisfies the following equation,

$$8S \cdot B \bmod l = (8 \cdot R + 8H(R, A, M) \cdot A) \bmod l$$

Verification without the cofactor 8 is a stronger way to verify a signature. Since the algorithm has good performance, ease of implementation, small key and signature sizes, it is rapidly being adopted in security of embedded devices also.

2.6 Software Requirements

MATLAB version 9.5 [R2018b].

Toolboxes required: Symbolic math toolbox, Communications Toolbox, Image Processing Toolbox.

III. RESULTS

Hash digest of 20-bit length is calculated by inbuilt MATLAB function. Next, the private key is to be given as input. Using this, the public key is computed. Value of R which is the ephemeral public key is obtained by computing abstract algebraic expression over Galois Field. Signature is got by calculating modulus involving Galois field and it is computed for image and video frames.

If the signature is found correct, then the same input is transmitted or else the input image is altered by the sender and then transmitted. Hence, it provides authentication for data.

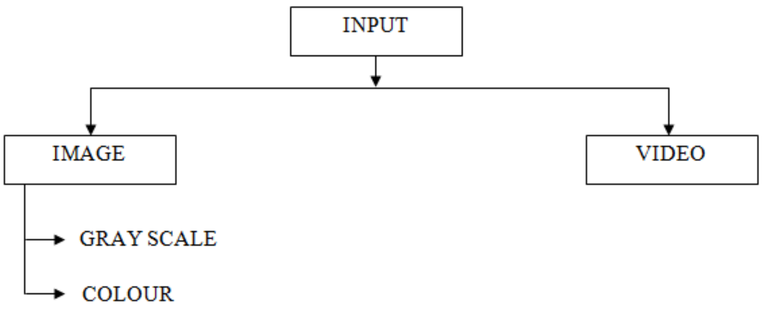

Figure 3.1 Types of multimedia data

In this digital signature scheme, signing and verification of generated signature has been performed for multimedia like image (color and gray scale) and video.

3.1 Image

Images are visual representations that do not have mobility. Using images, the content creator can convey information which can be more freely interpreted by the user. The unit of measurement used for computer graphics is the pixel. The term pixel is a contraction for picture element. A computer screen can be measured in pixels.
Here, both the input and output are images. During signature verification, if the signature is found incorrect then the user is not authenticated or registered user. Then the sender transmits the image after corrupting it. If the signature is found correct then the original input image is transmitted.

3.1.1 Gray Image

Here, gray image is taken as input

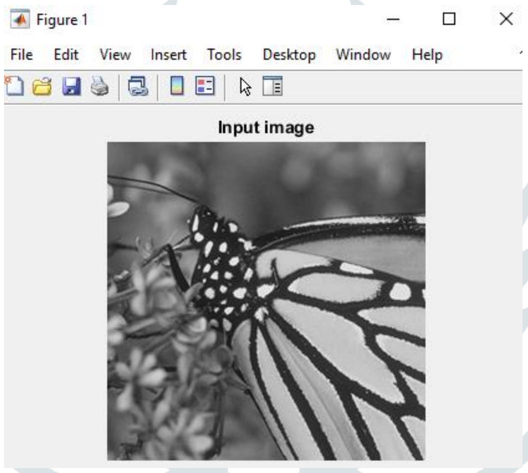

Figure 3.1.1.1 Input image (gray)

Case 1: When the signature is found correct, actual input image is transmitted.

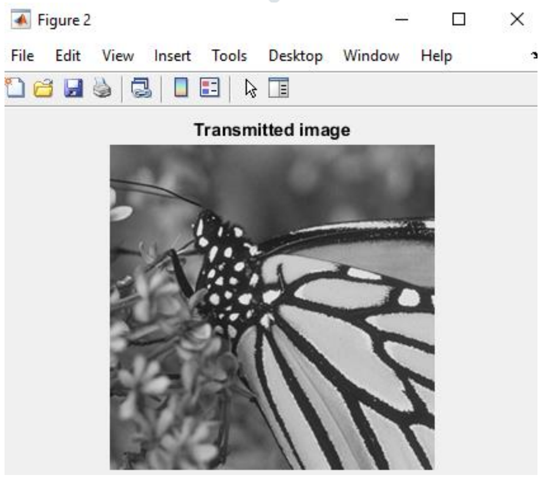

Figure 3.1.1.2 Transmitted image when the signature is correct

Case 2: When the signature is found incorrect, sender will corrupt the input image such that no unauthorized party can get to know the contents of the file.

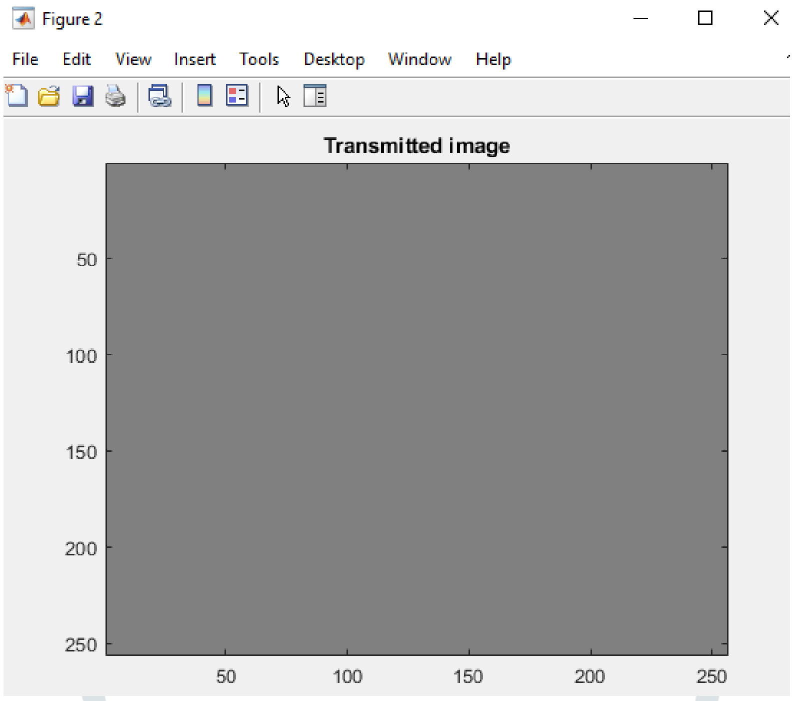

Figure 3.1.1.3 Transmitted image when the signature is not correct

3.1.2 Color Image

Here, color image is taken as input.

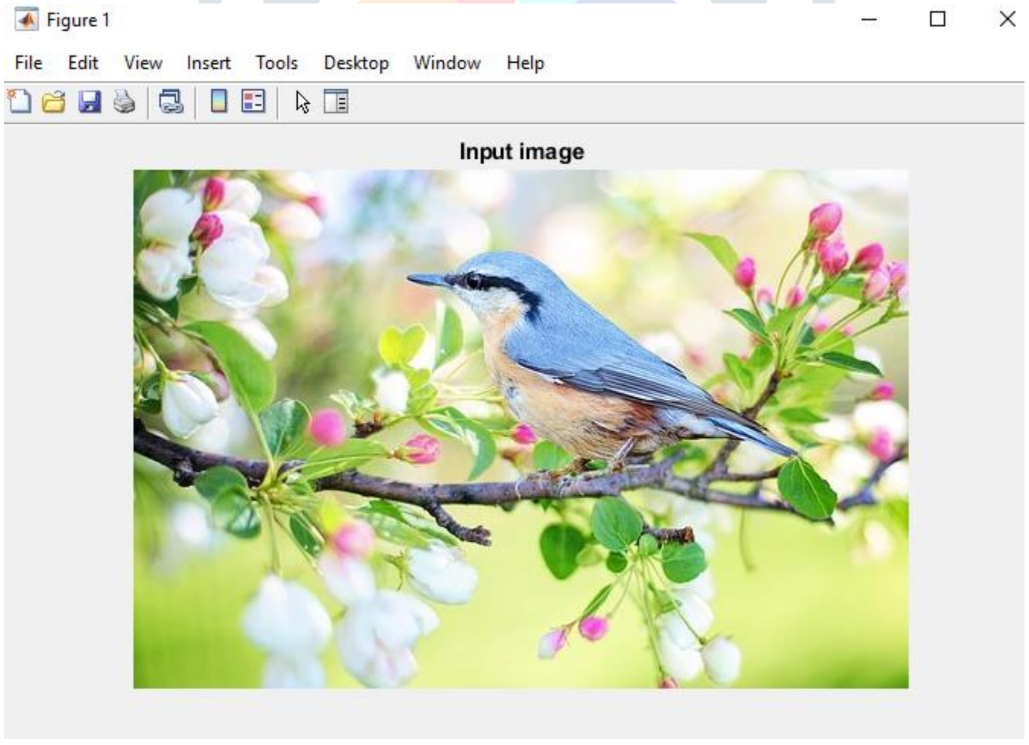

Figure 3.1.2.1 Input image (color)

Case 1: When the signature is found correct, actual input image is transmitted.

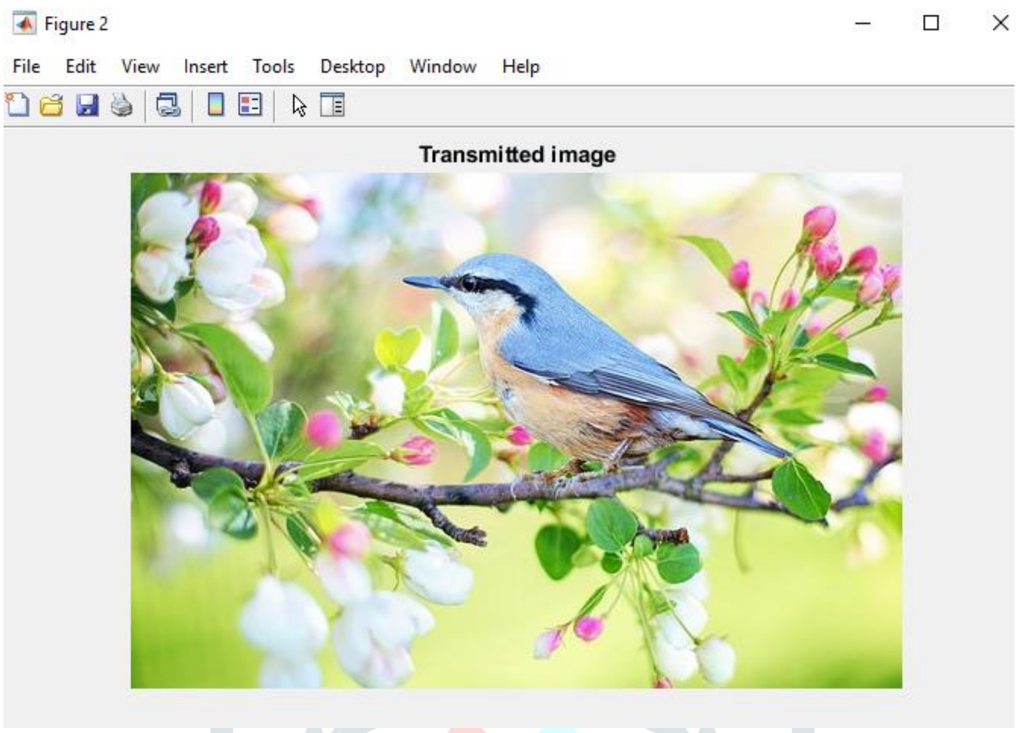

Figure 3.1.2.2 Transmitted image when the signature is correct

Case 2: When the signature is found incorrect, sender will corrupt the input image and then transmits it.

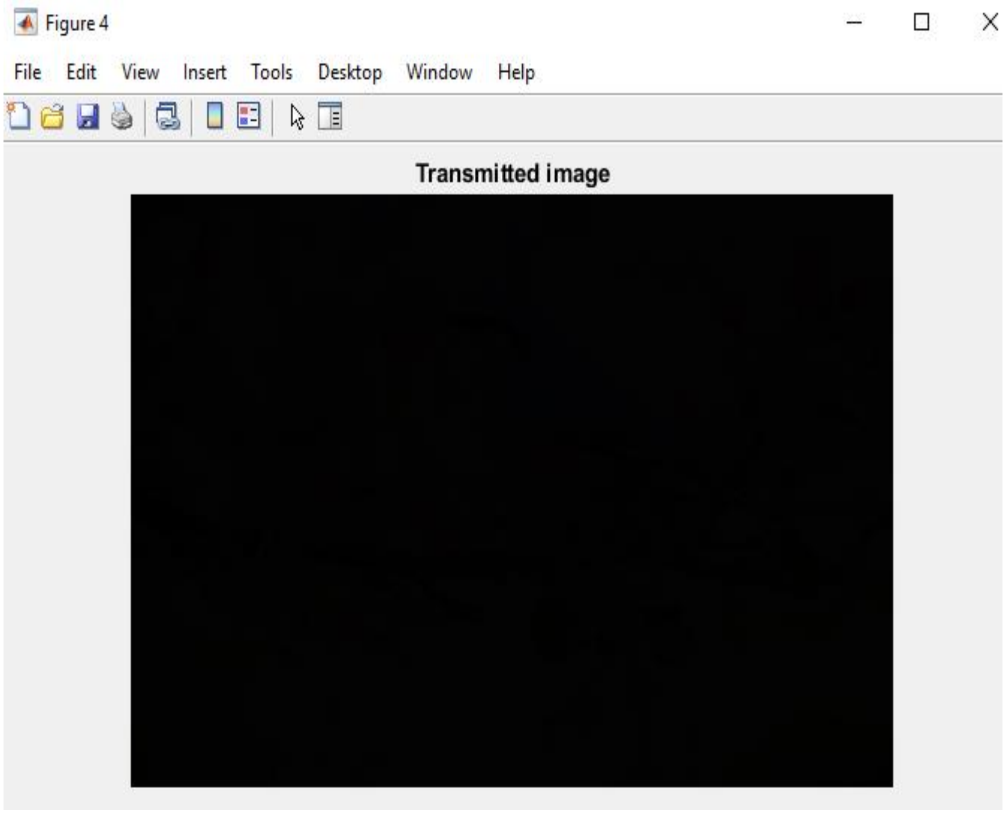

Figure 3.1.2.3 Transmitted image when the signature is not correct

3.1.3 Mean Square Error

The Mean Square Error (MSE) is the cumulative squared error between the altered and the original image. If the original and the transmitted image are same then the MSE between them will be zero. MSE is always a non-negative value. Closer value to zero represents less error. It is widely used in signal processing and image processing applications.

Table 1:MSE for image

IMAGE TYPE	MSE when signature is correct	MSE when signature is incorrect
Gray scale image	0	1.5563e+04
Colour image	0	3.6148e+04

3.2 Video

Here, input and output are in the form of video frames. A ten second video of mp4 format is considered as input. Signature is calculated for six continuous frames which shows some movement. During signature verification, if the signature is found incorrect then the user is not authenticated or registered user. Then the sender transmits the corrupted video frames. If the signature is found correct then the original video frames will be transmitted.

The selected six frames and the output obtained when the signature is correct and when it is incorrect is given as follows.

Frame 1

Frame 2

Frame 3

Frame 4

Frame 5

Frame 6

Figure 3.2.1 Input video frames

Case 1: When the signature is found correct, actual video frames are transmitted.

Frame 1

Frame 2

Frame 3

Frame 4

Frame 5

Frame 6

Figure 3.2.2 Transmitted frames when the signature is correct

Case 2: When the signature is found incorrect, sender will corrupt the video frames and then transmits it.

Frame 1

Frame 2

Frame 3

Frame 4

Frame 5

Frame 6

Figure 3.6.3 Transmitted frames when the signature is incorrect

3.3 EXECUTION TIME

Execution time is the time during which a program is running (executing), in contrast to other program life cycle phases such as compile time, link time and load time. In Matlab, a built-in function called tic-toc is used to estimate profiler time. Toc function counts the elapsed time from the stopwatch timer started by the tic function. It computes the internal time during the execution of the toc command, and displays the elapsed time upto the most recent call to the tic function that had no output, in seconds.
Execution time of computing signature for various multimedia input is shown in the below table:

Table 2: Execution time of signature calculation for various input

INPUT TYPE	SIZE	EXECUTION TIME (sec)
Text - Sentence	23 bytes	0.004233
Text – Document file	1.15 kB	0.004556
Image – Gray scale	64.6 kB	0.266309
Image - colour	42.4 kB	0.318870
Video	6 frames	1.876042

IV. DIFFERNCES BETWEEN PROPOSED AND EXISTING SYSTEM

DSA (Digital Signature Algorithm) is a FIPS (Federal Information Processing Standard) which is defined for digital signatures. It's security is dependent on a discrete logarithmic problem. When compared to RSA, DSA generates signature with high speed. But validation takes a longer time. Security can be broken if bad number generators are used.
ECDSA (Elliptical curve Digital Signature Algorithm) is an Elliptic Curve implementation of DSA (Digital Signature Algorithm). ECDSA was first proposed in 1992 by Scott Vanstone in response to NIST's (National Institute of Standards and Technology) request for public comments on their first proposal for DSS. It is able to provide the relatively the same level of security level as RSA with a smaller key.
ECDSA is a cryptographic scheme based on ECC. The Elliptic Curve Digital Signature Algorithm (ECDSA) is an elliptic curve analogue of the DSA. Signature computation is deterministic in EdDSA and its security is based on the intractability of some discrete logarithm problems. Thus, it can be concluded that, though key length is very small, it is safer than DSA and ECDSA which require high quality randomness for every signature.
In the proposed system, EdDSA is implemented over Galois Field. Hence, all the operations are different from normal arithmetic operations. Thus, it is more secure than the existing EdDSA system. Confidentiality and integrity for multimedia data like text, image and video also increases.
Several parameters of ECDSA and proposed EdDSA over Galois Field are compared and tabled in Table 3.

Table 3: Performance comparison of EdDSA with ECDSA

Parameters	ECDSA	Proposed EdDSA over GF(p^m)
Key length	384	10
Key generation (sec)	0.799	0.0006
Signature generation(sec)	0.0016	0.0002
Signature verification(sec)	0.0082	0.0007

IV.CONCLUSION

Nowadays, digital signatures are being used all over the Internet. Digital signature schemes are also used in electronic transactions over block chain. Several digital signature schemes like DSA, ECDSA are in existence.But they have limitations like large key length, high computational time for generation of signature, security compromisation. An algorithm to give a digital signature that authenticates multimedia data and provides non-repudiation is designed. It is faster than existing digital signature schemes and has a relatively small key length. Yet, it does not compromise on the data integrity and security it offers. Implementation of certain calculations over Galois field reduces computation time and size of the signature.

REFERENCES

[1] National Institute of Standards and Technology,1996, Entity Authentication using PublicCryptography, FIPS Publication,
[2] R. Housley, Vigil Security,August 2018,ISSN 2070-1721.

[3] IlariLiusvaara and Simon Josefsson, January 2017, Edwards-Curve Digital Signature Algorithm (EdDSA). RFC 8032.

[4] YolanRomailler, Sylvain Pelissier, 2017, Practical fault attack against the Ed25519 and EdDSA signature schemes, In Workshop on Fault Diagnosis and Tolerance in Cryptography.

[5] Sakshi Samaiya& Asst. Prof. Anupreksha Jain, An Implementation of GA based FPGA ALU unit,International Journal of Engineering Sciences & Research Technology, ISSN No. 2277-9655.

[6] Janardan Mahanta, Construction of Balanced Incomplete Block Design: An Application of Galois Field, Open Science Journal of Statistics and Application.

[7] Daniel J. Bernstein, Simon Josefsson, Tanja Lange, Peter Schwabe, and Bo-Yin Yang. 2015, EdDSA for more curves. Cryptology ePrint Archive, Report 2015/677.

[8] Daniel J. Bernstein, Peter Birkner, Marc Joye, Tanja Lange, and Christiane Peters.2008,Twisted edwards curves, in Cryptology ePrint Archive, Report 2008/013.

[9] Gregory Neven, Nigel P. Smart, Bogdan Warinschi.2009, Hash function requirements for Schnorr signatures, Journal of Mathematical Cryptology 3.

[10] ---(no editor), 17th annual symposium on foundations of computer science, IEEE Computer Society, Long Beach, California. MR 56:1766.

[11] ---(no editor), 2011, Proceedings of the 6th ACM symposium on information, computer and communications security, Hong Kong, March 22-24, 2011, Association for Computing Machinery. ISBN 978-1-4503-0564-8.

[12] Adrian Antipa, Daniel R. L. Brown, Robert P. Gallant.2006, Accelerated verifcation of ECDSA signatures, in SAC, 307-318. MR 2007d:94044.

[13] Daniel J. Bernstein, Tanja Lange,19 September 2011,eBACS: ECRYPT Benchmarking of Cryptographic Systems.

[14] Secure Hash Standard (SHS), March 2012, National Institute of Standards and Technology (NIST).

[15] Digital Signature Standard (DSS), 2013, National Institute of Standards and Technology (NIST).

[16] https://www.maximintegrated.com/en/app-notes/index.mvp/id/5767.