An Implementation of Combined Elliptic Curve Digital Signature Algorithm and LSB Watermarking for Image Authentication

Alifa Nurani Putri
Informatics/Computer Science, School of Electrical Engineering and Informatics
Institut Teknologi Bandung
Bandung, Indonesia
alifanuraniputri@gmail.com

***Abstract*—This paper propose an algorithm used to embed and verify digital signature for an image file especially photo. The algorithm combined ECDSA (*elliptic curve digital signature algorithm*) and LSB (*least significant bit*) watermarking. Digital signature can support three concepts in security: authenticity, integrity, and non-repudiation. The main point which will be explained in this paper are analysis of the proposed algorithm, solution design, and an implementation to provide the experiment scheme. Another one, this paper also try to explain the used of this algorithm in photo sharing social media for a better safety and extra security in the application layer.**

Keywords—cryptography; digital signature; ECDSA; watermarking; image processing;

I. Introduction

Nowadays with the growth of internet user, especially for social media, the number of piracy and ownership imputation is also rising up. One of them, which is often happened is photo piracy and imputation for enternainer and politic figures. This phemomenon can affect the popularity and cause bad image for them if they are not the uploader of those pohots. That phenomenon is one example of threat for user's content in social media. This kind of threat attack the authenticity and integrity of a photo upload by users.

In the Informatics field, one thing which can be a solution for that phenomenon is by providing a technology for embedding digital signature in the uploaded image. Social media especially photo sharing social media can provide this system to give a better safety for the users. Digital signature need a private key which is must be kept by the owner and the owner can uses it whenever uploading photo.

Digital Signature, which is a method in information security, used to protect a content in three security concepts, that are authenticity, integrity, and non-repudiation. Authenticity is a concept that guarantee the uploaded photo is truly uploaded by the user. Integrity means the photo that published is exactly the same as the photo when the user upload it. Non-repudiation means the user can't deny that he/she uploat that photo.

This paper will demonstrate an algorithm used to embed and verivy digital signature in an image or photo. The algorithm is a combination of ECDSA (*elliptic curve digital signature*) and LSB (*least significant bit*) watermarking. This combination can embed and verify digital signature in one file image without any additional file nor any tag. It can be used for providing a better security for photo sharing social media.

II. Literatures Study

A. Elliptic Curve Digital Signature Algorithm (ECDSA)

Digital signature is one of information security technique used to provide authenticity, non-repudiation, and integrity for a digital content. For example, a sender Alice sends a message to Bob. As receiver, Bob can prove that the message truly sent by Alice and Alice can't deny that she sent that message to Bob. Digital signature also can guarantee that the message which Bob received is exactly the same that Alice sent

There are some digital signature algorithms, one of them is *Elliptic Curve Digital Signature Algorithm* (ECDSA). ECDSA is a lightweight signature algorithm because it operates in elliptic curve group. ECDSA uses multiplication in elliptic curve, which is same as repeated additions of two points, as a basic operation. In general, there are three main components in every digital signature algorithm those are *key generation*, *signature generation*, and *signature verification*.

Key generation is the first thing to do when using digital signature. When a sender Alice sends a message to Bob, they must agree on a set of domain parameters of the elliptic curve that they will use later. As a sender, Alice must have a private key *dA* (a random value less than *n*, *n* is the order of the curve). Alice must keep the private key to herself. Alice also have a public key *QA* (the value of *QA* depends on the private key, QA = dA * G, G is a generator point). Alice can share her public key to the receiver for verifying.

When Alice sends a message, she will sign it with a function called signature generation. There are five steps in generating an ECDSA signature:

1. Calculate hash value of the message *m* with one hash algorithm, for example SHA1 (as used in this implementation) $e = HASH(m)$.
2. Find a random integer *k* in [*1*,*n-1*]
3. Calculate $r = x1 \pmod n$, where $(x1, y1) = k * G$ If $r = 0$ back to step 2.
4. Calculate $s = k - 1(e + dAr) \pmod n$. If $s = 0$ back to step 2.
5. The signature of a sender Alice is a pair <*r*,*s*>

When receiving a message, Bob can prove that the message comes from Alice and it was not altered during the sending process with signature verification. There are six steps:

1. Verify that the value of *r* and *s* are in [1,n − 1]. If not, the signature is invalid.
2. Calculate $e = HASH(m)$, with the same hash algorithm in signature generation
3. Calculate $w = s - 1 \pmod n$
4. Calculate $u1 = ew \pmod n$ and $u2 = rw \pmod n$
5. Calculate $(x1, y1) = u1G + u2QA$
6. Signature is valid if x1 = r(mod n), otherwise the signature is invalid.

B. LSB Watermarking

Watermarking is also used to protect a digital content especially media for being manipulated, edited, copied by unallowed parties with representing the ownership of it. There are severeal watermarks like text, logo, audio, binary, and others. Watermarking is a steganography based implementation.. Then, to detect what has been embedded, a watermark detector is used. The different between watermarking and a pure steganography is the content or media. In steganography, the content have no value, otherwise in watermarking the content have high value to be protected.

There are two ways to embed watermark in image by the human views, visible watermarking and invisible watermarking. Visible watermarking embed a watermark which can be detedted by human. Otherwise, invisible watermarking embed the watermark in a hidden way by human view, this kind of watermarking will be implemented in this paper.

There are also several methods for embedding watermark which are divided in spatial and transform. This paper use spatial method, which is a simple way to embed directly by changing the pixels' byte value of an image. This paper use a method called LSB (*least significant bit*) that edit the last bit value on one byte. By editing like that, the watermarked image is almost same as the original one, human can't detect it, it is invisible. Actually, LSB method is not a robust one because it is very easy to delete the watermark by editing all of the LSB, but LSB is a very simple one to embed watermark.

III. Proposed Analysis And Solution Designs

This paper propose an algorithm which is a combination of ECDSA (*elliptic curve digital signature algirithm*) and LSB (*least significant bit*) watermarking. The ECDSA is used to generate the signature by private key and content of the image, then also used to verify whether the image has been altered or not by using the valid public key. The LSB watermarking is used to embed that siganture to the image.

In this algorithm, digital signature can be embedded to the image in one file. So, no need of additional file to carry out the digital siganture. It is a simple way. The digital signature is embedded in the image body not in the header. It is supported by all standard image file in digital image processing.

A. ECDSA Generation and Verification design

To generate a pair of digital signature by using ECDSA, as what has been explained in section II, a message content is needed to be hashed. If the content is a string message or email, it can be used directly, but this paper will generate a signature for image. So, the first one is to generate a string to represent the image.

It is very simple to read an image file in the form of string. The problem is that the digital signature will be embedded in the image. So, after it has been ebedded, string of the image will change and it will affect the verification process. In the verification process the image should be hashed again and the signature valid if the hashed message is exactly the same with the hashed message in signature generation porcess. So that, it is a task in this proposed algorithm to design how that two hashed message will be same exactly.

To overcome this problem, actually a simple method can be used. ECDSA will generate a pair of sigantures that is <*r*,*s*>. The maximal length of it can be known by the domain parameters which have been agreed. Both *r* and *s* are integers in the modulo *n*, so the value is between 1 to *n*, because 0 is forbidden. With a determined maximal length of the digital signature, the maximal number of bits which will change in the image to embed/store the digital sognature can be determined too. So that, the trick is used by this proposed algorithm is by hashing the whole image in the string execpt a number of pixels which will be used to embed the digital signature.

For example an image which is 512x512 pixels. Each pixels is a 24 bit and represent three RGB colors, each 8 bit or 1 byte. Then an elliptic curve is agreed to be a base for ECDSA, for example using a NIST standard curve which is P-192 with the parameter n is 192 bit. So, the total size of digital signature is 384 bit, or sometimes it is added some character line '<', ',', and '>' to make the embedder work easier. So, to embed it in the image, about 384 bit or 96 bytes or 32 pixels is not used while hashing the image content. The complete steps to generate digital signature from an image with this proposed algorithm can be seen in Figure III.1.

When verifying the signature, the process is same. The hashed message is a string image ignoring the pixels that containt the digital signature. Other process is same as an original ECDSA verification algorithm.

B. *Watermark Embedder Design*

In the explanation before, a digital signature can be generated if the pixels which will be used to locate the signature is known, due to the signature size. The problem now is where is the location can be used to locate so the signature can be detected when when verifying. In this algortihm, the simplest way to embed a watermark is used, that is LSB (*least significant bit*).

Actually there several ways o locate it with the LSB. By using the determined of first pixel, or by locating in in a random pixels using determined seed. To make the implementation simple, this implementation used a determined first pixel, that is the first pixel. So, a signature with needs for example 32 pixels maximal size can be located in the 32 first pixels. When verifying, the size of signature can be known with the domain parameter, for example *n* pixels. So to get the signature is just reading the first *n* pixels LSB. This full steps is also explained in Figure III.1.

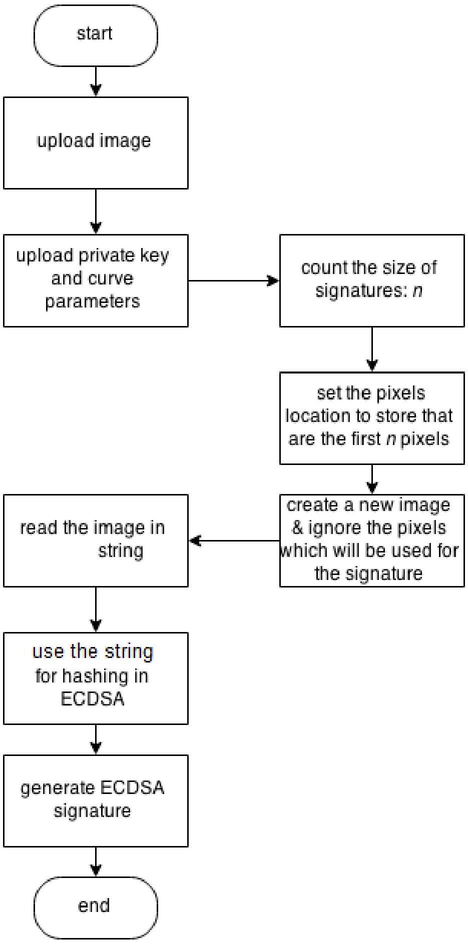

start
upload image
upload private key and curve parameters
count the size of signatures: *n*
set the pixels location to store that are the first *n* pixels
create a new image & ignore the pixels which will be used for the signature
read the image in string
use the string for hashing in ECDSA
generate ECDSA signature
end

Figure III-1 Generating Digital Signature Using the Proposed Algorithm

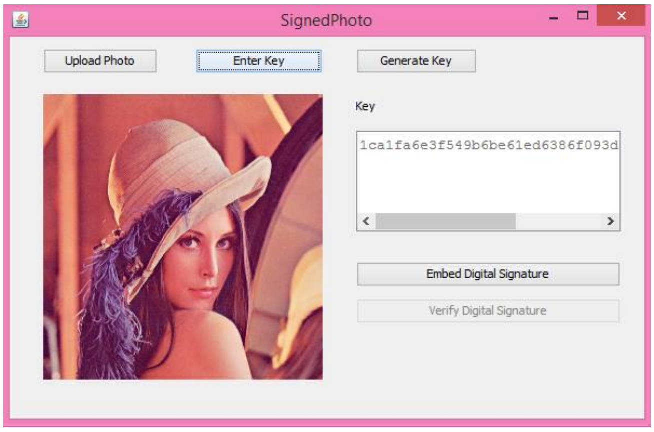
C. *Key Generation*

As what has been explained in section II, ECDSA (*Elliptic Curve Digital Signature Algorithm*) actually needs some domain parameters which must be agreed first. This requirement rise one problem, which is to define an elliptic curve and to choose one base point in that curve is not easy for user. Besides that, not all elliptic curve are safe. To overcome this problem, in this implementation key generation menu that allow user to choose a NIST standard curve and parameter (i.e. P-192, P-256) which can be selected in the key generation menu. Each standard curve has standard domain parameters which is safe to be used for generating and verifying digital signature. User can save the keys and can be used anytime.

IV. Implementation And Experiments

A. *Implementation and Evinronment*

This paper implements the proposed algorithm as an digital signature embedded for image, and the reverse process to verify it. The language used by this implementation is JAVA, with a Java Swing graphical user interfase. For manipulate the image, Java DIP (*digital image processin*g) is used with a `BufferedImage` class which is used to handle and manipulate the image data. There are two main user interfaces, for key generation and for the embedder system. Figure IV.1 is the main user interface, for generate the key user can select the 'Generate Key' button. User can Embed the digital signature after enter a private key and can verify the photo after enter a public key the system automaticly detect the key entered.



Key

1ca1fa6e3f549b6be61ed6386f093d

Figure IV-1 Main User Interface

B. *Experiment Scheme*

By embedding digital signature, three concepts in security is covered. Those are authenticity, integrity, and non-repudiation. So that, some scenarios below is done to show the work of proposed algorithm to do those three concepts.

1. A valid one, this is the normal flow.
2. The private key used when embedding the digital signature is not the pair of public key used to verify the image. This scheme could happen in the real life as the uploader who is an hijacker don't know the private key the attecked user.
3. The image is manipulated and the digital signature is safe.

All of those experiments use a standard image for digital image processing named *lenna* which can be seen in Figure V.1. This image is a bitmap image with 512x512 pixels and 768 kB. For the ECDSA, this experiment use a P-192 NIST standard curve with the parameter explained in Table IV.1.

Figure IV-2 Standard Image for Experiments

Table IV.1 NIST P-192 parameters

Parameter	Value
p	fffffffffffffffffffffffffffffffeffffffffffffffff
a	fffffffffffffffffffffffffffffffefffffffffffffffc
b	64210519e59c80e70fa7e9ab72243049feb8deecc146b9b1
xG	188da80eb03090f67cbf20eb43a18800f4ff0afd82ff1012
yG	07192b95ffc8da78631011ed6b24cdd573f977a11e794811
n	ffffffffffffffffffffffff99def836146bc9b1b4d22831

Then, for this experiments, the true pair private and public is on the Table IV.2

Table IV.2 Keys for Experiment

private	7020724359194037196468500719509698087436148528638434647 75
public	(22569331822019466207629855432536116965112690389850057060 47, 26563994452501462004609924025211704109239975334784577265 50)

V. Experiment Results and Analysis

A. Case 1: A Valid One

This case use a pair private and public key, and the image is not altered until the image verified.

Tabel V-1

Embed the Digital Signature	
Hash value	71726ea18fbe71c523a8cfc08d043bf7b4823c83
ECDSA <filename>@<*r*>@<*s*>@	alifa.bmp@522247abb897fcbadceef2527b907fbd9917cd79aca72d63@459cdbb1d4f838cab3cb5b69adb1f79d1964a605e2ee1ffb@
Verify	
Hash value	71726ea18fbe71c523a8cfc08d043bf7b4823c83
ECDSA detected <filename>@<*r*>@<*s*>@	alifa.bmp@522247abb897fcbadceef2527b907fbd9917cd79aca72d63@459cdbb1d4f838cab3cb5b69adb1f79d1964a605e2ee1ffb@
Status: Valid	

This case just show how embedding a digital signature can prove the ownership if the the public and private key is a pair and the content is original. This is the normal case, the default case if nothing bad happened to the image.

B. Case 2: Private key is not a Pair with the Public Key

This case use a private key which is not a pair with the public key. So, for this case the private key is: 266932896617528939807845910177034192412968419750624112592 1.

Tabel V-2

Embed the Digital Signature	
Hash value	71726ea18fbe71c523a8cfc08d043bf7b4823c83
ECDSA <filename>@<*r*>@<*s*>@	alifa.bmp@bb820262c6b2774dba214c970fcbb8ac6910f788fd605b4c@d923a3f8317d96a33cc4a53a96d33336e224d1f359aa6bdd@
Verify	
Hash value	71726ea18fbe71c523a8cfc08d043bf7b4823c83
ECDSA detected <filename>@<*r*>@<*s*>@	alifa.bmp@bb820262c6b2774dba214c970fcbb8ac6910f788fd605b4c@d923a3f8317d96a33cc4a53a96d33336e224d1f359aa6bdd@
Status: Invalid	

This case show that if the uploader use wrong private key because he/she is a hijacker and the private is kept safety by the user, people can know that it is a photo/image uploaded by other people. because the verification process is show an invalid result. This verification uses user's public key which is open to everyone. The verification only valid when the private key is a pair with the public key.

C. Case 3: The image is manipulated and the digital signature is safe

This case will affect the hash value because the image, in Figure V.1, has been manipulated. The hash value is different, so the ECDSA is invalid.

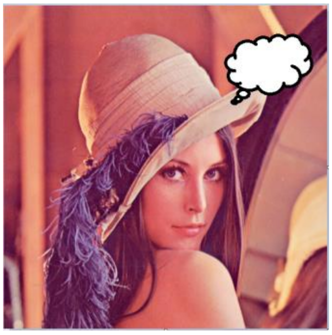


Figure V-1 Altered Image

Tabel V-3

Embed the Digital Signature	
Hash value	71726ea18fbe71c523a8cfc08d043bf7b4823c83
ECDSA <filename>@<*r*>@<*s*>@	alifa.bmp@f36bdc8ff5a9dd37b8c5d9c27b6325bbcd298aeb8214e7d4@a4196bdd518245cf12b0a2366258247cd3a6029f75883bc3@
Verify	
Hash value	152ccc26156051bf8de9c344482090fe77d0bdbe
ECDSA detected <filename>@<*r*>@<*s*>@	alifa.bmp@5af08e68ed4bdb94607e7acc1f993c61ce0b7cb7b76ccaf0@c55fa1be7e048409513f40eae80d27ed6fd34dda74e20c8@
Status: Invalid	

The third case show that if any person manipulate the image, and the signature pixels still safe, it can be detected because the verification will also invalid. This is happened because ECDSA use a hash function to protect the content, and if any change, the verification will invalid.

VI. Case Study: Photo Sharing Social Media

This system of embedding a digital signature to image can be used in photo sharing social media (i.e. Instagram). This system can protect the value of user's content that is integrity, to authenticate the ownership of photo uploaded, and also for non-repudiation, so no user can deny what content that he/she uploaded before, it will used if a content have a problem later. So, the social media can use it as an obligation to set a digital signature anytime the user upload a photo.

The design to use digital signature in social media is by giving a secret private key to all user when registering. Each user has their own private key, and he/she must keep it. Then, all of the public key are open to public. The private key is the most precious value for each user.

With the implementation of digital signature in photo sharing social media, it can provide a better security, that are for authenticity, integrity, and non-repudiation. It will give a great value for that social media with this nowadays situation in the social media that many people can become a hijacker for several motivations, although it is just for fun. Embedding digital signature in every uploaded photos could be a solution for this situation

VII. Conclusions

These are the conclusions for this paper proposed algorithm and implementation:

1. Digital signature can be used to protect image in three security concepts, that are: authenticity, integrity, and non-repudiation.
2. This proposed algorithm is a simple way to generate, embed, detect, and verify digital signature for all standard image types, and no other additional file is needed.
3. This implementation can be developed by using the safer hash function and more complex watermark embedder to give a better security.

Acknowledgment

With the completion of author in writing this paper, the author want to express praise for the Almighty Allah SWT, Who gives blessing during this preparation to submit this paper. The author also want to thank Mr. Rinaldi as author lecturer in IF4020 Cryptography who gives her some knowledges used to propose this paper. Also for all friends that help the author direct or indirectly.

References

[1] Anoop MS, Elliptic Curve Cryptography An Implementation Guide

[2] Rinaldi, Bahan Kuliah IF3058 Kriptografi: Digital Watermarking

[3] http://www.tutorialspoint.com/java_dip/ accessed on May, 10th 2015 at 6:48 pm

Pernyataan

I hearby declare that this paper is my own writing, no addaptation, or translation of others' paper, and not plagiarism.

Bandung, May 11 2015

Alifa Nurani Putri - 13511074