

International Journal of Advance Research in Computer Science and Management Studies

Research Article / Survey Paper / Case Study

Available online at: www.ijarcsms.com

Advanced Security Algorithm Using QRCode™ Implemented for an Android Smartphone System: A_QR

Sayantana Majumdar¹

Department of Computer Science,
St. Xavier's College, Kolkata, India

Abhisek Maiti²

Department of Computer Science,
St. Xavier's College, Kolkata, India

Biswarup Bhattacharyya³

Department of Computer Science,
St. Xavier's College, Kolkata, India

Dr. Asoke Nath⁴

Department of Computer Science,
St. Xavier's College, Kolkata, India

Abstract: Due to tremendous growth of media and communication technology, now it is a real challenge to share/send some information through the insecure network/media ensuring that, the receiver will get the authentic information. For this reason Nath et al developed an information security system combining Cryptography & Mobile Computing together, and presented a method 'A_QR'(Authentic QR). In the present paper, the authors present a new scheme to digitally sign any small file and encode it to a QR Code to widely share the information over the network/media. Android Smartphones can be used to quickly decode the QR Code™ obtaining the stored data and verify the authenticity of the decoded information. Quick Response Code(QR Code™) are a machine-readable two dimensional matrix barcodes used for encoding information. Recently, it has become very popular due to its fast readability and greater storage capacity compared to standard UPC barcodes. Digital signature is a mathematical scheme for demonstrating the authenticity of a digital message or document. Digital signatures are used in the cases where it is important to detect forgery or tampering. Here, first the information is digitally signed using ECDSA algorithm, then encoded into a QR Code. A simple android app is developed in order to obtain the information from the QR Code and to check the authenticity of the decoded information. ECDSA is a modern and extremely secure crypto-algorithm which is currently internationally accepted for online transactions & many other policies. ECDSA is free from plain text attack or any brute force attack. The present method may be used to publish non-modifiable sensitive public document like mark sheets and certificates.

Keywords: QR Code, ECDSA, Digital signature, Android, Cryptography

I. INTRODUCTION

To maintain the authenticity of the file over the media & network, extremely secure standard cryptographic method must be applied to verify and prohibit the modification of the information. Here the authors have used 256-bit ECDSA^[5] encryption with prime256v1^[12] curve and SHA-256^[4], which is fully compliant with NIST 800-57^[5] security standard and FIPS 186-3^[13] security standard. QR Code™ is very flexible, popular and widely used. QR Code™ can be shared not only over the digital media but also over any visual or print media. The proposed scheme may be used to certify any printed or digital document.

As with Elliptic curve cryptography (ECC), ECDSA(Elliptic Curve Digital Signature Algorithm) is an approach to public-key cryptography based on the algebraic structure of elliptic curves over finite fields.

A digital signature is an authentication mechanism that enables the creator of message to attach a code that acts as a signature. A digital signature scheme typically consists of three algorithms:

- » A key generation algorithm that selects a private key uniformly at random. The algorithm outputs the private key and a corresponding public key.

- » A signing algorithm that, given a message and a private key, produces a signature.
- » A signature verifying algorithm that, given a message, public key and a signature, either accepts or rejects the message's claim to authenticity.

In our method to secure information we use following procedure:

1. Generate a private key and its corresponding public key using ECDSA with prime256v1, which is a NIST recommended named curve for elliptic curve cryptography and write the Base64^[16] encoded private key and public key to “privkey.txt” and “pubkey.txt” respectively.
2. Generate the digital signature by signing a small input file with the private key and using SHA-256 as the hash algorithm. Write the Base64 encoded digital signature to “sig”.
3. For size compression and unification, compress the original file along with the corresponding digital signature into a zip file.
4. Generate QR Code™ for the newly created zip file.

To decode the information using Android device, we use the following procedure:

1. Store “pubkey.txt” in the android application package(APK) every time a new private and public key pair is generated and then compile the android app.
2. Scan the QR Code™ using the camera of the device and write the decoded data into a new zip file.
3. Extract the contents of the zip file and then delete the zip file after successful extraction.
4. If requested by the user, verify the authenticity of the main file using corresponding Base64 decoded “sig” and “pubkey.txt” and display the verification result.

II. ALGORITHM

a) Key pair Generation Algorithm: (GenKeys)

STEP 1: Start

STEP 2: Create a Key Pair^[2]

STEP 3: Initialize the Key Pair randomly

STEP 4: Generate the Private Key and Public Key

STEP 5: Write Base64 encoded private and public key to “privkey.txt” and “pubkey.txt”

STEP 6: End

b) Digital Signature Generation Algorithm: (GenSig)

STEP 1: Start

STEP 2: Get a Signature Object^[2]

STEP 3: Decode the Base64 encoded “privkey.txt”

STEP 4: Initialize the Signature Object with decoded private key

STEP 5: While read buffer data from input file $\neq 0$

Update the data to be signed by each byte read.

STEP 6: Generate the Digital Signature

STEP 7: Write the Base64 encoded digital signature into 'sig' file

STEP 8: End

c) Digital Signature Verification Algorithm: (VerSig)

STEP 1: Start

STEP 2: Import “pubkey.txt”, “sig” and the file to be verified(infile)

STEP 3: Decode the Base64 encoded public key and obtain key specification

STEP 4: Generate a public key object from the obtained key specification

STEP 5: Decode the Base64 encoded digital signature and store it in a byte array

STEP 6: Initialize a Signature object and sign it with the newly generated public key

STEP 7: While read buffer data from infile $\neq 0$

 Update the data to be verified by each byte read.

STEP 8: Generate a new digital signature

STEP 9: If imported signature is equal to the generated signature then

 Display 'Verification: True'

 else

 Display 'Verification: False'

STEP 10: End

d) QRCode Generation Algorithm: (GenQR)

STEP 1: Start

STEP 2: Import the source file(infile)

STEP 3: Call GenSig(infile)

STEP 4: Compress “sig” and infile into “result.zip” file

STEP 5: Create an empty string data

STEP 6: Convert “result.zip” into a Base64 encoded string and store in “data”

STEP 7: Input the image format and resolution of the QR Code to be generated

STEP 8: Input Error Correction Level

STEP 9: Using zxing^[1] library method convert 'data' into a BitMatrix object 'bitmatrix'

STEP 10: Write bitmatrix to an image

STEP 11: End

N.B- BitMatrix represents a 2D matrix of bits.

III. CONCEPTUAL MODELS

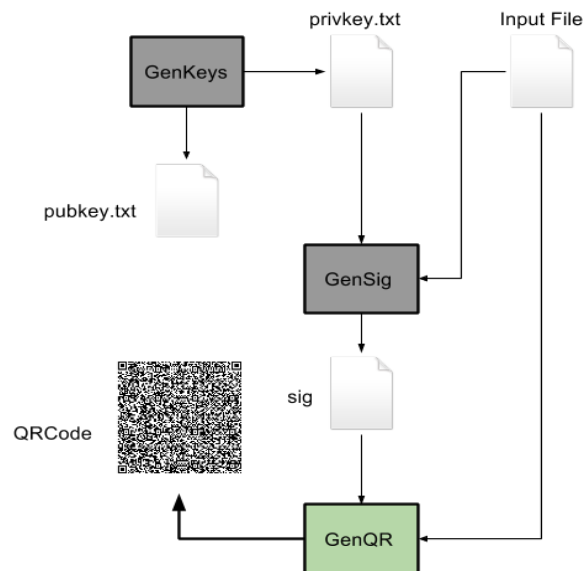
a) *Encoding:*

Figure 1 Encoding scheme model

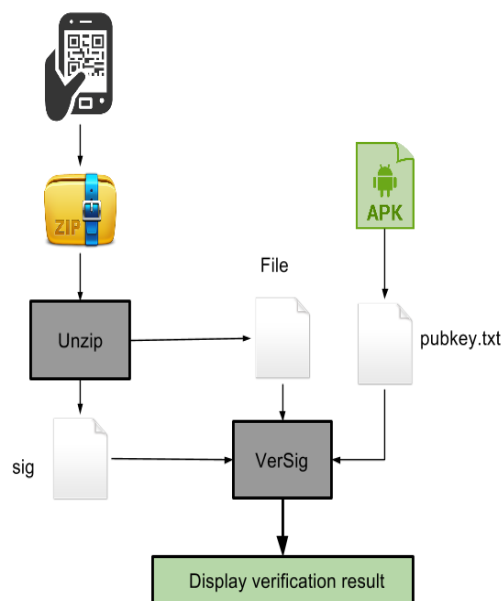
b) *Decoding*

Figure 2 Decoding scheme model

Above diagrams show the conceptual model of the project. Digitally signing the file and generation of QR Code will be done in traditional desktop environment whereas decoding the QR Code and verification system are developed to work in Android mobile environment.

IV. RESULTS AND DISCUSSION

a) CASE #1: With an HTML File

1. Original File:

COMPUTER Sc (HONS) 5th. SEMESTER EXAMINATION, YEAR 2014						
NAME: JOHNNY APPLESEED Regd. No: A01-1112-0760-12 Roll: 3-15-12-0540 Current Year: 3						
Subject Code	Subject Name	Max Marks	C.A.	E.E.	Agg	Result
CMSA3505	Computer Graphics, Software Engineering, Microprocessor and Design & Analysis of Algorithms	100	16	64	80	PASS
CMSA3555	MATLAB, Advanced DBMS & Microprocessor Applications	100	20	72	92	PASS

SEMESTER-V RESULT

Figure 3 HTML file

2. Base64 encoded Signature:

MEQCIGfgCYCm6Ny4zafozo3q/S6LBQJLsgx2wD988+fnstTUAiB7FqdWp45IXdI4DcgiNEuSXA7vpL49/aFvLesL0m7a
+Q==

3. QRCode:

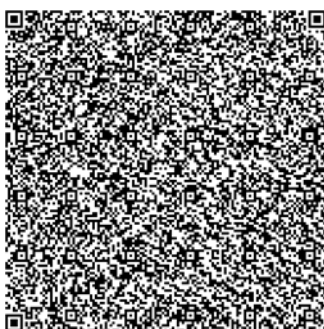


Figure 4 QRCode image

4. After Decoding:

COMPUTER Sc (HONS) 5th. SEMESTER EXAMINATION, YEAR 2014						
NAME: JOHNNY APPLESEED Regd. No: A01-1112-0760-12 Roll: 3-15-12-0540 Current Year: 3						
Subject Code	Subject Name	Max Marks	C.A.	E.E.	Agg	Result
CMSA3505	Computer Graphics, Software Engineering, Microprocessor and Design & Analysis of Algorithms	100	16	64	80	PASS
CMSA3555	MATLAB, Advanced DBMS & Microprocessor Applications	100	20	72	92	PASS

SEMESTER-V RESULT

Figure 5 Decoded file

b) CASE #2: With a PDF file

1. Original File:



St. Xavier's College (Autonomous)
30, Mother Teresa Sarani, Kolkata - 700016

SEMESTER MARKSHEET

COMPUTER Sc. (HONS.) 5th. SEMESTER EXAMINATION, YEAR 2014

The following is the statement of marks obtained by **SAYANTAN MAJUMDAR**

Roll : 3-15-12-0535

Regd. No.: A01-1112-0755-12

at the aforesaid Examination held in **NOV - DEC 2014**

Paper Code	Paper Title	FM	QM	CA	SE	Total	Agg (TH+PR)	Grade (Point)	Credit Earned
CMSA3555	PR MATLAB, ADVANCED RDBMS , AND MICROPROCESSOR APPLICATIONS	100	40	15	70	85	85	A+ (9)	4
CMSA3505	TH COMPUTER GRAPHICS, SOFTWARE ENGINEERING, MICROPROCESSOR AND DESIGN & ANALYSIS OF ALGORITHMS	100	40	15	61	76	76	A (8)	14
Total Credits Earned :									18

SGPA = 8.22

Abbreviation Codes:

FM: Full Marks QM: Qualifying Marks CA: Continuous Assessment SE: Semester-End Exam Agg: Aggregate Marks

TH: Theory PR: Practical DB: Debarred AB: Absent RW: Result Withheld INC: Result Incomplete RA: Reported Against

A. C. Gomes

Controller of Examinations

ST. XAVIER'S COLLEGE (AUTONOMOUS)
KOLKATA

DATE : 30/03/2015

Disclaimer:

St. Xavier's College (Autonomous) is not responsible for any inadvertent error that may have crept in the results being published on this website. The results being published on this website are for immediate information to the examinees.

Fig. 6 PDF file

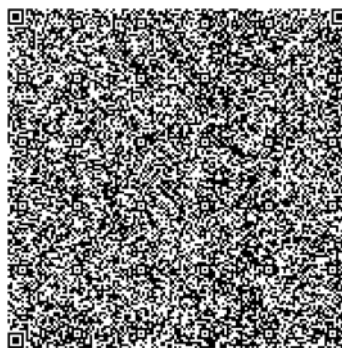
2. Note 1: PDF files are generally quite large. So this pdf file was converted into a text file.

3. Text file:


The following is the statement of marks obtained by SAYANTAN MAJUMDAR
 Roll : 315120535 Regd. No.: A011112075512
 at the aforesaid Examination held in NOV DEC 2014
 A. C. Gomes
 Controller of Examinations
 ST. XAVIER'S COLLEGE (AUTONOMOUS)
 KOLKATA DATE : 30/03/2015
 St. Xavier's College (Autonomous)
 30, Mother Teresa Sarani, Kolkata 700016
 SEMESTER MARKSHEET
 COMPUTER Sc. (HONS.) 5th. SEMESTER EXAMINATION, YEAR 2014
 Paper Code Paper Title FM QM CA SE Total Agg
 (TH+PR)
 Grade
 (Point)
 Credit
 Earned
 CMSA3555 PR MATLAB, ADVANCED RDBMS, AND
 MICROPROCESSOR APPLICATIONS
 100 40 15 70 85 85 A+ (9) 4
 CMSA3505 TH
 COMPUTER GRAPHICS, SOFTWARE ENGINEERING,
 MICROPROCESSOR AND DESIGN & ANALYSIS OF
 ALGORITHMS
 100 40 15 61 76 76 A (8) 14
 Total Credits Earned : 18
 SGPA = 8.22
 Abbreviation Codes:
 FM: Full Marks QM: Qualifying Marks CA: Continuous Assessment SE: SemesterEnd Exam Agg: Aggregate Marks
 TH: Theory PR: Practical DB: Debarred AB: Absent RW: Result Withheld INC: Result Incomplete RA: Reported Against
 Disclaimer:
 St. Xavier's College (Autonomous) is not responsible for any inadvertent error that may have crept in the results
 being published on this website. The results being published on this website are for immediate information to
 the examinees.

*Fig.7 – Text file screenshot***4. Base64 encoded Signature:**

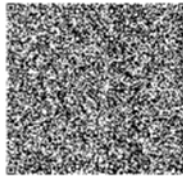
MEQCIBtn4vUHi+WDJEUbxmF2Y8AqmmheFK2YmmNSKgsyymZAiB5+g3B5LG7zBQr7i46zXuuQnC/nSLI+tEUwCt
 gEK0mXQ==

5. QRCode:*Fig. 8 QRCode image*

6. **Note 2:** This QRCode can be embedded in the original pdf file(to generate a digitally signed mark sheet in this case) as follows:



St. Xavier's College (Autonomous)
30, Mother Teresa Sarani, Kolkata - 700016



SEMESTER MARKSHEET

COMPUTER Sc. (HONS.) 5th. SEMESTER EXAMINATION, YEAR 2014

The following is the statement of marks obtained by **SAYANTAN MAJUMDAR**
Roll : 3-15-12-0535 **Regd. No.: A01-1112-0755-12**
 at the aforesaid Examination held in **NOV - DEC 2014**

Paper Code	Paper Title	FM	QM	CA	SE	Total	Agg (TH+PR)	Grade (Point)	Credit Earned
CMSA3555 PR	MATLAB, ADVANCED RDBMS , AND MICROPROCESSOR APPLICATIONS	100	40	15	70	85	85	A+ (9)	4
CMSA3505 TH	COMPUTER GRAPHICS, SOFTWARE ENGINEERING, MICROPROCESSOR AND DESIGN & ANALYSIS OF ALGORITHMS	100	40	15	61	76	76	A (8)	14
Total Credits Earned :									18

SGPA = 8.22

Abbreviation Codes:
 FM: Full Marks QM: Qualifying Marks CA: Continuous Assessment SE: Semester-End Exam Agg: Aggregate Marks
 TH: Theory PR: Practical DB: Debarred AB: Absent RW: Result Withheld INC: Result Incomplete RA: Reported Against

A. C. Gomes
 Controller of Examinations
 ST. XAVIER'S COLLEGE (AUTONOMOUS)
 KOLKATA

DATE : 30/03/2015

Disclaimer:
 St. Xavier's College (Autonomous) is not responsible for any inadvertent error that may have crept in the results being published on this website. The results being published on this website are for immediate information to the examinees.

Fig. 9 Digitally signed marksheet

- c) **CASE #3: With an SVG image:**

1. **Original image:**



Fig. 10 - SVG image

2. Base64 encoded Signature:

MEUCIH57UcOr2P2gN/rQmtwkJU28NLp/n2bn52kFLXnz2/KZAiEA1T4rqQ6vVjAPd0gTXyh4BwAwkvPqJOaxojgzQ
CeA9E=

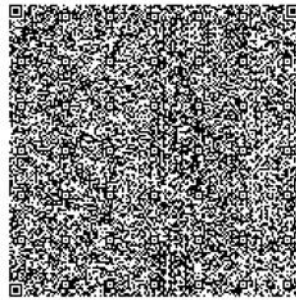
3. QRCode:

Fig. 11- QRCode image

4. After Decoding:

Fig.12-Decoded SVG image

d) CASE #4: With a Plain Text File:**1. Original File Contents:**

“I define UNIX as 30 definitions of regular expressions living under one roof.” Donald Knuth, Digital Typography, ch. 33, p. 649 (1999)

2. Base64 encoded Signature:

MEQCIE3q9ghPggbx1/nzM5FPKqjPAwQ/csWraQ8Dt+/eWuvAiASzP0qXKV5fYoOciYZiuIKjAgDVuJewmc4boN
IX0CE6w==

3. QR Code

Fig. 13 QRCode image

1. After Decoding:

"I define UNIX as 30 definitions of regular expressions living under one roof." Donald Knuth, Digital Typography, ch. 33, p. 649 (1999)

2. Verification:

If we modify the data of the decoded file to

Input: *I define LINUX as 30 definitions of regular expressions living under one roof."*

Donald Knuth, Digital Typography, ch. 33, p. 649 (1999)

Verification Result: **False**

If we try the verification with more minor change, such as change '*Donald Knuth*' to '**Donald Knuth**' i.e.

Input: *"I define UNIX as 30 definitions of regular expressions living under one roof."*

Donald Knuth, Digital Typography, ch. 33, p. 649 (1999)

Verification Result: **False**

- 3. Note:** In every test case, the digital signature has been generated accordingly as "sig" but the public key is generated once as "pubkey.txt".

The Base64 encoded public key used for the above test cases is:

MFkwEwYHKoZIzj0CAQYIKoZIzj0DAQcDQGAEpgs1hcBSov7QJ0MJlft8Mwi284CxG2j8jsrhD8lI2FNR2jxCXC9t
hKpI+5PFxzTnwUIUpkBMbC02sIpK90mW9A==

V. CONCLUSION AND FUTURE SCOPE

In cryptography the process of securing data has been improved drastically in last few years. In our proposed method we use 256-bit ECDSA encryption to digitally sign the file. The primary benefit promised by ECC is a smaller key size, reducing storage and transmission requirements, i.e. that an elliptic curve group could provide the same level of security afforded by an RSA-based system with a large modulus and correspondingly larger key: for example, a 256-bit ECC public key should provide comparable security to a 3072-bit RSA^[6] public key.

The hardest ECC scheme (publicly) broken to date had a 112-bit key for the prime field case and a 109-bit key for the binary field case. For the prime field case this was broken in July 2009 using a cluster of over 200 PlayStation 3 game consoles and could have been finished in 3.5 months using this cluster when running continuously^[7]. For the binary field case, it was broken in April 2004 using 2600 computers for 17 months^[11]. In August 2013, it was revealed that bugs in some implementations of the Java class SecureRandom^[2] sometimes generated collisions in the key value. This allowed a solution of the private key, in turn allowing stealing bitcoins from the containing wallet on Android app implementations, which use Java and rely on ECDSA to authenticate transactions.^[14]

From the concept of Quantum Computer, it can theoretically break the ECDSA or in general the ECC cryptosystem using a modified Shor's Algorithm for solving the discrete logarithm problem on elliptic curves^[8]. A quantum computer to attack elliptic curve cryptography can be less than half the size of a quantum computer to break an equivalently classically secure version of RSA. The work of Proos and Zalka show how a quantum computer to break 2048-bit RSA requires roughly 4096 qubits while a quantum computer to break the equivalently secure 224-bit ECC requires between 1300 and 1600 qubits. Depending on the growth rate of quantum computers in the future, elliptic curve cryptosystems may become attackable by a quantum computer many years before an equivalently secure RSA scheme^[9]. But practically building a Quantum Computer is

not yet possible. Elliptic curve cryptography is also prone to side-channel attacks like differential fault attacks^[10] and power attacks.

Although ECDSA is an extremely secure cryptosystem till date and it will remain secure for a long while, the scenario may change quickly with the fast growth of technology and science. It is clear that, to maintain the security of this scheme we must update the crypto-system regularly. From the practical aspect, this may be a challenge to this project because of the compatibility issues.

Another major limitation of this scheme is the storage capacity of the QR Code™. In terms of capacity QR Code™ is better than UPC (Universal Product Code) barcodes. Still, even with low error detection level it's not enough to store file greater than ~4.2KB containing alphanumeric characters only. Though it is not impossible to split a large file and generate more than one QR Code™, it is not always feasible or recommended.

Last major limitation is limited resource and processing power of mobile devices. Scanning large QR Code™ with low error detection level in different lighting environments require a good built-in camera, which may not be available with every Android devices. Moreover, an updated security algorithm may require more complex processing than a mobile device can handle.

References

1. Zxing Library: <https://github.com/zxing/zxing/>
2. Java security libraries: <http://docs.oracle.com/javase/7/docs/api/java/security/package-summary.html>
3. Android Libraries: <https://developer.android.com/reference/packages.html>
4. Secure Hash Standard: <http://csrc.nist.gov/publications/fips/fips180-4/fips-180-4.pdf>
5. NIST Standard: <http://csrc.nist.gov/publications/nistpubs/800-57/>
6. RSA reference: <http://people.csail.mit.edu/rivest/Rsapaper.pdf>
7. http://lcal.epfl.ch/112bit_prime
8. Michael A. Nielsen; Isaac L. Chuang (9 December 2010). Quantum Computation and Quantum Information: 10th Anniversary Edition. Cambridge University Press. pp. 202–. ISBN 978-1-139-49548-6.
9. Proos, John; Zalka. "Shor's discrete logarithm quantum algorithm for elliptic curves". QIC. arXiv:quantph/0301141. Archived from the original on 2004. Retrieved 3 May 2014.
10. Biehl, Ingrid; Meyer, Bernd; Müller, Volker (2000). "Differential Fault Attacks on Elliptic Curve Cryptosystems". Advances in Cryptology – CRYPTO 2000. Lecture Notes in Computer Science 1880: 131–146. doi:10.1007/3-540-44598-6_8. ISBN 978-3-540-67907-3.
11. <https://www.certicom.com/news-releases/300-solution-required-team-of-mathematicians-2600-computers-and-17-months->
12. <http://csrc.nist.gov/groups/ST/toolkit/documents/dss/NISTReCur.pdf>
13. FIPS 186-3 Standard: http://csrc.nist.gov/publications/fips/fips186-3/fips_186-3.pdf
14. <https://bitcoin.org/en/alert/2013-08-11-android>
15. Sayantan Majumdar, Abhisek Maiti, Biswarup Bhattacharyya, Asoke Nath, "A new encrypted Data hiding algorithm inside a QR Code™ implemented for an Android Smartphone system: S_QR algorithm". International Journal of Innovative Research in Advanced Engineering(IJIRAE), ISSN: 2349-2163, Issue 4, Volume 2 (April 2015) (<http://www.ijirae.com/volumes/Vol2/iss4/07.APAE10093.pdf>)
16. Base64: <https://www.ietf.org/rfc/rfc3548.txt>