International Journal of Engineering & Technology, 7 (2.7) (2018) 758-762

International Journal of Engineering & Technology

Website: www.sciencepubco.com/index.php/IJET

Research Paper

On using Aryabhatta Remainder Theorem to Decrypt a Message with RPrime and Rebalanced RSA

Ch. J.L. Padmaja[1]*, V.S.Bhagavan[2], B.Srinivas[3]

[1]Research scholar, Koneru Lakshmaiah Education Foundation, Vaddeswaram, India
[2] Koneru Lakshmaiah Education Foundation, Vaddeswaram, India
[3]Department of Technical Education, Andhra Pradesh, India
**Email: padmajachivukula@gmail.com*

Abstract

RSA is the most world widely used asymmetric cryptosystem for network transactions. Through this article, we propose a new implementation of Aryabhatta Remainder theorem (ART) in place of the existing Chinese Remainder Theorem (CRT) to solve congruencies in the decryption phase for the faster variants of RSA such as RPrime RSA and Rebalanced RSA. Through our observations, we prove that using ART for CRT has improved the overall decryption speed of RPrime and Rebalanced RSA.

Keywords: *Aryabhatta remainder theorem; Chinese remainder theorem; Rebalance; RPrime; Rebalanced.*

1. Introduction

Digital and electronics payments for secure transactions use cryptosystems which encrypt, decrypt and sign a message. These researchers developing these security intensive applications are always in search of a faster and better cryptosystems. So, we find many faster and better variants of the original RSA cryptosystem proposed by Rivest, Shamir and Adleman in 1997 [7]. Popularly known faster variants of RSA are Batch RSA[2], MultiPower RSA [9], Multiprime RSA, Rebalanced RSA [12], RPrime RSA [5]. All of them use Chinese remainder Theorem (CRT) to solve congruencies while encrypting and decrypting messages, and CRT is prone to few threats. T.R.N Rao and Hang [6] suggested the use of a remainder theorem built by Aryabhatta, known as Aryabhatta Remainder Theorem (ART). The authors suggest that ART is much faster than the CRT, there are no known attacks on ART with RSA cryptosystems.

Hence, we chose to replace the CRT with Aryabhatta Remainder Theorem, wherever applicable. Through our observation, we have found that ART cannot be applied to Batch RSA[2] and Multipower RSA[1]. A.Singh [8] in his Master thesis has provided an implementation of ART with Multiprime RSA and through his finding suggested that ART+ Multiprime RSA performs better than CRT + MultiPrime RSA[1]. Through our article, we suggest the implementation of ART for Rebalanced RSA and RPrime RSA. The following sections detail the difference between CRT and ART, implementation of Rebalanced RSA and RPrime RSA with ART and their performance comparison.

2. Solving Congruencies CRT Vs ART

Chinese Remainder theorem was proposed by a Chinese mathematician in 13th A.D. to solve the congruencies of first order with one unknown [10]. The Chinese remainder solution to congruencies is shown in the table 1 and 2.

Table 1: Algorithm CRT to solve for decryption exponent in key generation stage

Solving congruencies using for d'=CRT(v1,v2; u1,u2; Z)
$Z=\Pi^{t}_{i=1}$ ui, g.c.d (ui, uj) = 1, $\forall i \neq j$ yi = $(Z/ui)^{-1}$ mod pi d' = $[\Sigma^{t}_{i=1}$ vi (Z/ui) yi] mod Z d = 2d'+a

The only difference between these two is we need to reconstruct the decryption exponent d from d' using the equation d=2d+a. The algorithm mentioned in table 1 is used during the key generation process to obtain d. Algorithm in table 2 is used while decrypting the Cipher and generate the message in the decryption stage.

Table 2: Algorithm CRT to solve M during the decryption stage

Solving congruencies to generate Message M=CRT(Mp1,Mp2; p1,p2; N)
$Z=\Pi^{t}_{i-1}$ pi, g.c.d (pi, pj) = 1, $\forall i \neq j$ yi = $(N/pi)^{-1}$ mod pi M = $[\Sigma^{t}_{i=1}$ vi (N/pi) yi] mod N

From the above equations, we can observe that the number of modular inverse operations is directly proportional to the number of primes. The more number of primes, the more inverse operations.

Aryabhatta Remainder theorem works on this issue i.e., to reduce the number of inverse operations to solve the residues problem. The table below shows the algorithm of Aryabhatta Remainder Theorem suggested by. T.R.N Rao and Hang [6].

Table 3: ART Algorithm to solve M during the decryption stage

Solving the Congruencies using ART for M M=**ART (**Mp1,Mp2,Mp3..Mpk; p1,p2,p3..pk; N**)**

Begin
Initialize P => $\prod^{t}_{i=1}$ pi, such that gcd (pi , pj) => 1 $\forall i \neq j$,
Initialize Loop Variables $N_1 \leftarrow 1$, $M_1 \leftarrow$ Mp1
Begin Loop
Begin for i from 2 to t do the following:
$N_i \leftarrow N_i\text{-}1 \cdot p_i\text{-}1$
Ci ← Ni^{-1} mod mi (also denote | Ni^{-1} | mi))
ui ← [(vi - xi-1) · Ci] mod p i
Mi ← Mi -1 + ui · Ni
End for
End Loop
Output : Return M
End

From the ART algorithm we can clearly observe that the number of modular inverses are not directly proportional to the number of prime and hence requires lesser number of modulo inverse operations, when compared with CRT. This is the advantage of ART over CRT. The following section provides the implementation of the above algorithm on Rebalanced RSA[12].

3. Rebalanced RSA

Rebalanced RSA[12] works on improving the weakness of decryption exponent d suggested by Weiner [12]. In this method, focus is shifted from improving encryption process to fasten up the decryption process.

There are three stages i.e., key generation stage to generate the public and private keys; encryption stage to encrypt the message into a cipher and the decryption stage to decrypt the message from the cipher. The figure 1 shows the processes involved in encrypting by a sender and decrypting the message by the receiver using Rebalanced RSA.

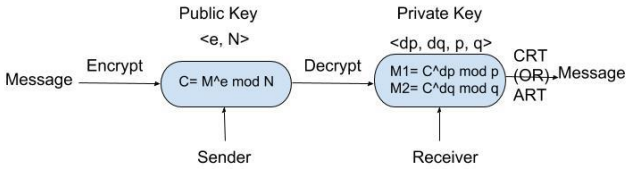


Fig. 1: Rebalanced RSA encryption and Decryption

Key generation : Generates two keys; public key <N, e> and private key <dp,dq,p,q>

- Where, N is product of two random primes p,q i.e., N=p.q with each prime of bit length log(n/2) bits such that the g.c.d (p-1,q-1) =2
- Select two random integers dp and dq such that g.c.d (dp,p-1) and g.c.d (dq, q-1) =1 and dp = dq mod 2
- Calculate two variants of d, such that d ≡dp mod p-1 and d ≡dp mod p-1 Apply CRT to solve the congruences.
- Obtain e using e=d^{-1} mod $\phi(N)$

Encryption : Generate cipher C from Message M

- Choose any integer M, a cipher can be generated using the Traditional RSA's [1] encryption process i.e., C = M^e mod N

Decryption : Generate Message M from cipher C

For each pair of <dp, p> and <dq,q> Generate a M using the equation

$Mp=C^{dp} \bmod p$
$Mq=C^{dq} \bmod q$

M can be obtained by solving the above congruencies. In the following section, examples were provided for decryption using CRT and ART.

The below section details the process of key generation, encryption and decryption of Rebalanced RSA using CRT and ART. Given Message (M)= 17, p1=7 & p2 =5, d1= 5 and d2=7.

Table 4:Key Generation and Encryption of Rebalanced RSA

Key Generation -
p1=7, p2=5; N=p1 . p2=35 ; $\phi(N) = (p1-1).(p2-1)$ =24
dp1=5, dp2=7; a=dp mod 2 =1
v1=(dp1-a)/2=2; u1=(p1-1)/2=3; d1'= v1 mod u1= 2 v2=(dp2-a)/2=3; u2=(p2-1)/2=2; d2'= v2 mod u2= 1
Solving the Congruencies using CRT
Z=u1 . u2 = 6; d'=CRT(2,3;3,2;6)
$y1= (6/3)^{-1} \bmod 3= 2^{-1} \bmod 3 =2$ $y2= (6/2)^{-1} \bmod 2= 3^{-1} \bmod 2 = 1$ d'= (2.2.2+3.3.1) mod 6 =(17) mod 6= 5 d= 2d'+a=11
e=d^{-1} mod $\phi(N)$=11^{-1} mod 24=11
Public Key <N, e>= <35, 11>
Private Key <p1, p2, dp1, dp2,>= <7, 5, 5, 7>
Encryption - Generating the Cipher C
Message M = 17
C= M^e mod N= 17^{11} mod 35 = 33
Decryption - Generating Message from Cipher
$Mp1=C^{dp1} \bmod p1 =33^5 \bmod 7=3$ $Mp2=C^{dp2} \bmod p2 = 33^7 \bmod 5=2$

3.1. Decryption with CRT

To decrypt the Message M from Mp1 and Mp2 we applied the CRT algorithm specified in Table 2. Cipher C= 33 is decrypted and Message M=17 is extracted as shown in the table 5.

Table 5: Decrypting Message M using CRT with Rebalanced RSA

Solving the Congruencies using CRT for M
M=CRT(**33^5, 33^7; 7, 5; 35)**
$y1= (35/7)^{-1} \bmod 7= 5^{-1} \bmod 7 = 3$ $y2= (35/5)^{-1} \bmod 5= 7^{-1} \bmod 5 = 3$
M= $[(33^5.5.3)+(33^7.7.3)] \bmod 35$ = 895574333412 mod 35 = 17

3.2. Decryption with ART

The Message M is decrypted from the cipher using Mp1 and Mp2 on which the ART algorithm specified in Table 3. The table 6 shows the decryption of Message M which requires only one modular inverse operation, whereas CRT needs two modulo inverse operations.

Table 6: Decrypting Message M using ART Rebalanced RSA

Solving the Congruencies using ART for M M=**ART (**3, 2; 7, 5; 35**)**					
i	**Ni**	**|Ni|pi**	**Ci**	**Ui**	**Mi**
1	1	-	-	-	3
2	1.7=7	|7|5=2	$\|7^{-1}\|5=3$	|(2-3).3|5=2	3+2.7=**17**

The following section provides the implementation of the above algorithm on RPrime RSA.

4. RPrime RSA

C.A.M Paxiao introduced RPrime RSA [5, 6], which is a combination of Multi prime RSA and Rebalanced RSA, i.e., it used more than two primes and also uses multiple decryption exponents dp1, dp2, dp3...dpk etc. There are no known security attacks on this variant of RSA making it more secure than the other faster variants. Also, it has three stages, key generation which is based on Rebalanced RSA; Encryption and decryption are similar to Multi Prime RSA. The figure 2 shows the processes involved in encrypting by a sender and decrypting the message by the receiver using RPrime RSA.

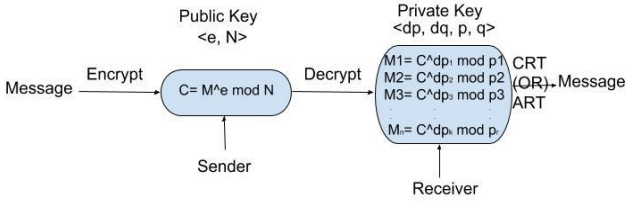


Fig. 2: RPrime RSA encryption and Decryption

Key generation : Generates two keys; public key **<N, e>** and private key <dp1,dp2,dpk..dpr;p1,p2,p3..pk>

- Where, N is product of multiple random primes p1,p2,p3..pk i.e., $N=\Pi^{t}_{i-1}$ pi with each prime of bit length log(k/2) bits such that the g.c.d(p_1-1,..,p_k-1) =2
- Here, g.c.d(dp1,p_1-1) ≡ g.c.d(dp2, p_2-1) ≡ g.c.d (dp_k, p_k-1)=1 and dp1 ≡ dp2 mod 2 ≡ dpr mod 2
- Calculate the variants of d, such that d' ≡dp1 mod p_1-1 .. d' ≡ dpk mod p_k-1 Apply CRT to solve the congruencies.
- Obtain e using $e=d^{-1} \bmod \phi(N)$ which is similar to the Rebalanced RSA.

Encryption : Generate cipher C from Message M

- This encryption stage as mentioned earlier is similar to the Rebalanced RSA. Choose any integer M, a cipher can be generated using i.e., **$C = M^e \bmod N$**

Decryption : Generate Message M from cipher C

This decryption is similar to the decryption process of Multi Prime RSA but with more pairs of primes and decryption exponents. For each pair of <dpk, pk> Generate a M using the equation

$Mp1=C^{dp1} \bmod p1$
$Mp2=C^{dp2} \bmod p2$
. . .
. . .
$Mpk=C^{dpk} \bmod pk$

M can be obtained by solving the above congruencies. In the following section, examples were provided for decryption using CRT and ART.

The table below shows the key generation, encryption and decryption for a Message using 3 primes.
Given M = 73, p1= 3, p2 = 5, p3 = 7, ,dp1 = 95, dp2 = 89 and dp3 = 59.

4.1. Decryption with CRT

Message M is obtained from Mp1 and Mp2. We applied the CRT algorithm specified in Table 2. Cipher C= 103 is decrypted and Message M=73 is extracted as shown in the table 8.

Table 7: Key Generation and Encryption of RPrime RSA

Key Generation - Public Key <N, e>
p1=3, p2=5, p3=7; N = p1 . p2 . p3 = 105 ; $\phi(N) = (p1-1).(p2-1).(p3-1) = 48$
dp1=95, dp2=89, dp3=59; a=dp mod 2 =1
v1=(dp1-a)/2=47, u1=(p1-1)/2=1; d1'= v1 mod u1= 47 v2=(dp2-a)/2=44, u2=(p2-1)/2=2; d2'= v2 mod u2= 0 v3=(dp3-a)/2=29, u3=(p3-1)/2=3; d3'= v3 mod u3= 2
Solving the Congruencies using CRT
Z=u1 . u2 . u3 = 6; d'=CRT(47, 0, 2; 1, 2, 3; 6)
$y1= (6/1)^{-1} \bmod 1= 6^{-1} \bmod 1 = 0$ $y2= (6/2)^{-1} \bmod 2= 3^{-1} \bmod 2 = 1$ $y3= (6/3)^{-1} \bmod 3= 2^{-1} \bmod 3 = 2$ d' = (47.6.0+44.3.1+29.2.2) mod 6 = (0+132+116) mod 6 = 248 mod 6 => 2 d = 2d'+a=5
$e=d^{-1} \bmod \phi(N)=5^{-1} \bmod 48=29$
Public Key <N, e>= <105, 29>
Private Key<p1, p2, p3, dp1, dp2, dp3>=<3,5,7,95,89,59>
Encryption - Generating the Cipher C
Message M = 73
$C= M^e \bmod N= 73^{29} \bmod 105 = 103$
Decryption - Generating Message from Cipher
$Mp1=C^{dp1} \bmod p1 = 103^{95} \bmod 3=1$ $Mp2=C^{dp2} \bmod p2 = 103^{89} \bmod 5=3$ $Mp3=C^{dp3} \bmod p3 = 103^{59} \bmod 7=3$

Table 8: Decrypting Message M using CRT with RPrime RSA

Solving the Congruencies using CRT for M
M=**CRT(**103^{95}, 103^{89}, 103^{59}; 3, 5, 7; 105**)**
$y1= (105/3)^{-1} \bmod 7= 35^{-1} \bmod 1 = 2$ $y2= (105/5)^{-1} \bmod 7= 21^{-1} \bmod 2 = 1$ $y3= (105/7)^{-1} \bmod 7= 15^{-1} \bmod 3 = 1$
M= [(103^{95}.35.2)+(103^{89}.21.1)+(103^{59}.15.1)] mod 105 = 73

4.2. Decryption with ART

The decryption using ART with RPrime RSA is similar to the decryption of Rebalanced RSA but for more number of primes.

Though the number of primes are increased, the number of modular inverse operations required by the ART algorithm are less than that of the CRT. The generation of Message M using ART is shown in table 9.

Table 9: Decrypting Message M using ART with RPrime RSA

M=**ART(**1, 3, 3; 3, 5, 7; 105**)**					
i	**Ni**	**|Ni|pi**	**Ci**	**Ui**	**Mi**
1	1	-	-	-	1
2	1.3=3	|3|5=3	$\|3^{-1}\|5$=2	|(3-1).2|5=4	1+4.3=13
3	3.5=15	|15|7=8	$\|15^{-1}\|7$=1	|(3-13).1|7=4	13+4.15=73

Note: In the key generation process of both the Rebalanced RSA and RPrime RSA, we have used CRT to obtain the decryption exponent as ART. ART is only used in the decryption stage.

5. Implementation

We have implemented the ART and CRT for large modulo sizes such as a 4096 bit on system with Macintosh Sierra Operating system with 16 GB RAM. We chose to implement ART in java as many predefined classes are provided in JDK with support of RSA cryptosystem [11]. We used one such feature called the BIGINTEGER class. To test our implementation, we have used the Credit card Data set provided by DataTrans [10]. The Message (M)

consists of Card Number, Expiry date without slashes or hyphens, and CVV shown in the table below.

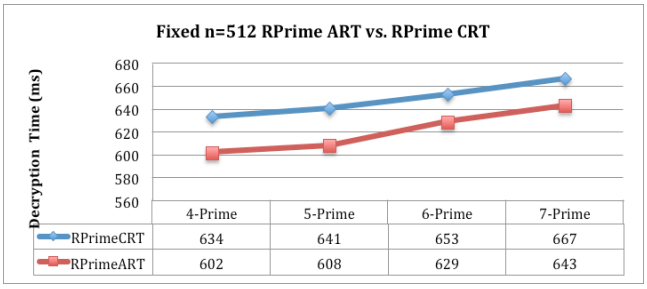
Table 10: Formulation of Message M from Credit Card data.

Name	Akimoto Sakimura
Type	MasterCard
Number	3569990010030442
Expiry Date	122018
CVV	123
Country Code	JPN
Message	**356999001003044212201812 3**

22 such message were encrypted and decrypted with RPrime CRT/ART and Rebalanced CRT/ART. The average of the decryption times was calculated. We have presented a comparison of the decryption speeds and the relative performance analysis in the following section.

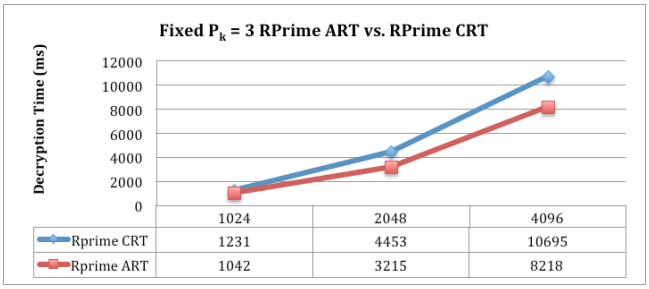
6. Performance of CRT Vs ART

Two comparisons were made due to the nature of RPrime and RSA[5] having variations in the number of primes (p_1, p_2,.., p_k). We have tested our implementation on number of primes ranging from 4 to 7 primes with a constant modulo size of 512, shown in Graph 1.



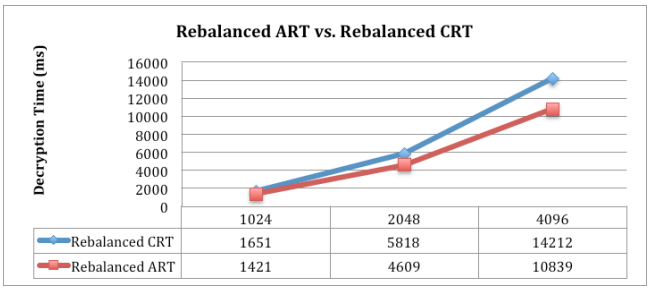
Graph 1: Fixed bit length and varying number of primes with Decryption times RPrime RSA.

The second set of comparisons was made against fixed number of primes, i.e., we chose 3 primes, shown in Graph 2. Decryption speed is measured in microseconds. A comparison was made on RPrime CRT[2] and RPrime ART with 1024, 2048 and 4096 size moduli.



Graph 2: Varying bit length and Decryption Time RPrime RSA

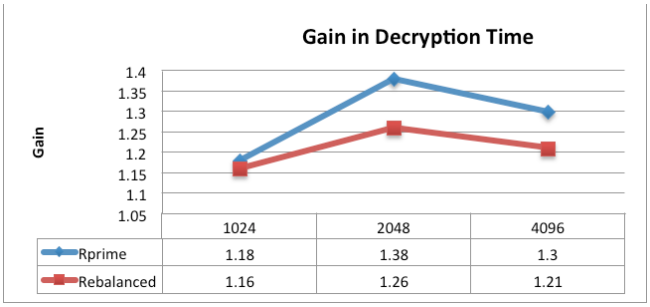
Rebalanced RSA uses only 2 primes and a comparison is made on the varying bit lengths of 1024, 2048 and 4096 bits as shown in Graph 2.



Graph 3: Varying bit length and Decryption Time of Rebalanced RSA

RPrime RSA and Rebalanced RSA with ART are compared (shown in graph 4) by their decryption time gains. The gain in decryption time is calculated using the following equation,

$$\text{Gain} = \frac{Decryption\ time\ of\ RPrime\ for\ a\ particular\ bit\ length}{Decryption\ time\ of\ Rebalanced\ for\ a\ particular\ bit\ length}$$



Graph 4: Gain in the Decryption time of RPrime and Rebalanced RSA

Note: Here there are two primes in Rebalanced RSA and three primes for RPrime RSA (pk=3).

From our comparisons we can observe that both RPrime RSA and Rebalanced RSA when implementing ART are faster than their CRT counterparts. RPrime RSA exhibits a better speed gain than Rebalanced RSA. Both RPRime RSA and Rebalanced RSA have a lesser speed gain with larger bit lengths such as 4096.

7. Conclusion

We have implemented Aryabhatta Remainder Theorem (ART) on RPrime RSA and Rebalanced RSA. A comparison of the performance is made with varying bit length and fixed bit lengths. Also, we have tested the implementation up to 7 primes. Using ART in place of CRT improved the decryption speeds of both Rebalanced and RPrime RSA. RPrime RSA shows a slightly better speed in decrypting, as there are no known threats or attacks on this faster variant, and hence is the better of the two.

References

[1] Boneh D, Shacham H., "Fast variants of RSA". CryptoBytes, Vol. 5, No.1, pp. 1-9, (2001).
[2] Fiat A, "Batch RSA", *Advances in Cryptology: Proceedings of Crypto '89*, Vol. 435, pp. 175–185, (1989).
[3] Knuth DE, "*The Art of Computer Programming – Volume 2: Seminumerical Algorithms*", Addison Wesley, (1969).
[4] Paixao CAM, "Implementacao e analise comparativa de variacoes do criptossistema RSA" Master's thesis, Inst. de Matematica e Estatıstica, Univ. de Sao Paulo., (2003).
[5] Paixao CAM., "An efficient variant of the RSA cryptosystem", *IACR Cryptology ePrint Archive*, pp.159, (2003).
[6] Rao TRN, Yang CH, "Aryabhatta Remainder Theorem: Relevance to Public-key Crypto-algorithms", *Circuits, Systems and Signal Processing*, Vol.25, No.1, (2006).
[7] Rivest R, Shamir A, Adleman L, "A method for obtaining digital signatures and public-key cryptosystems", *Commun. ACM*, Vol.21, No.2, pp.120–126, (1978)

[8] Singh A, “Improving the RSA Crypto Computations”, Master Thesis, University of Louisiana, (2006).
[9] Takagi T, “Fast RSA-type Cryptosystem Modulo p k q”, In H. Krawczyk, ed., Proceedings of Crypto ’98, 1462 of LNCS, Springer-Verlag, pp. 318–326, (1998).
[10] WebLink to DataTrans Credit Card Data https://www.datatrans.ch/showcase/test-cc-numbers
[11] WebLink to tutorial of RSA BiInteger Implementation using Java http://www.java2s.com/Code/Java/Security/SimpleRSApublickeyencryptionalgorithmimplementation.htm
[12] Wiener MJ, “Cryptanalysis of short RSA secret exponents”, *IEEE Transactions on Information Theory*, Vol.36, No.3, pp. 553–558, (1990).