Received December 5, 2019, accepted December 23, 2019, date of publication January 17, 2020, date of current version February 11, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.2967396

Towards a Novel Generalized Chinese Remainder Algorithm for Extended Rabin Cryptosystem

JUSTIN ZHAN[1], PETER J. SHIUE[2], SHEN C. HUANG[2], AND BENJAMIN J. LOWE[1]
[1]Department of Computer Science and Computer Engineering, University of Arkansas, Fayetteville, AR 72701, USA
[2]Department of Mathematical Sciences, University of Nevada Las Vegas, Las Vegas, NV 89119, USA

Corresponding author: Justin Zhan (jzhan@uark.edu)

This work was supported in part by the United States Department of Defense under Grant W911NF1910482, and in part by Arkansas Research Alliance under Grant DN145250.

ABSTRACT This paper proposes a number of theorems and algorithms for the Chinese Remainder Theorem, which is used to solve a system of linear congruences, and the extended Rabin cryptosystem, which accepts a key composed of an arbitrary finite number of distinct primes. This paper further proposes methods to relax the condition on the primes with trade-offs in the time complexity. The proposed algorithms can be used to provide ciphertext indistinguishability. Finally, this paper conducts extensive experimental analysis on six large data sets. The experimental results show that the proposed algorithms are asymptotically tight to the existing decryption algorithm in the Rabin cryptosystem with the key composed of two distinct primes while maintaining increased generality.

INDEX TERMS Algorithms, cryptography, Rabin cryptosystem.

I. INTRODUCTION

The Chinese Remainder Theorem was first proposed by Sunzi in the third century, although the complete theorem was given in year 1247 by Qin [1]. The theorem has been widely applied in many fields including cryptography, signal processing, vehicular technology, communications, etc. [2], [3], [4], [5], [6].

This paper proposes a number of theorems and algorithms to deal with the extended Rabin cryptosystem and quadratic residue problem, which is formally stated as follows: Let $\mathbb{N}$ be the set of all positive integers. The quadratic residue problem is to find x, where $x^2 \equiv a \pmod{n}$, $a, n \in \mathbb{N}$. Solving this problem involves a certain amount of nontrivial mathematics. When n is the product of two distinct large prime numbers, the quadratic residue problem is a key challenge in public key cryptography. Computing the square root of quadratic residues has applications to Elliptic Curve cryptography [7] and the Rabin cryptosystem [8]. In the Rabin cryptosystem, plain-text is encrypted by modular exponentiation. The decryption process requires taking the square root of a quadratic residue in two distinct prime modular spaces and combining the result.

The classical Rabin cryptosystem only considers a product of two distinct large primes, p and q, to be its public key n, where $n = pq$ [8]. The classical Rabin cryptosystem lacks theoretical generality. The Rabin cryptosystem can be extended through the algorithms proposed in this paper. The extended Rabin cryptosystem is allowed a public key n composed of an arbitrary finite number of distinct primes raised to any natural number powers, $n = p_1^{t_1} p_2^{t_2} \cdots p_k^{t_k}$. This increases the generality of the cryptosystem. Further, the classical Rabin cryptosystem does not provide ciphertext indistinguishability. That is, if an eavesdropper knows the public key n and intercepts the ciphertext c, then performing encryption on a candidate message m can reveal if c encodes m. The proposed extended Rabin cryptosystem can be used to overcome this limitation.

This paper solves the quadratic residue problem modulo n, where $n = p_1^{t_1} p_2^{t_2} \cdots p_k^{t_k}$, using the proposed algorithms. Furthermore, a generalized algorithm to compute the 2^k solutions from the proposed method is developed. The theoretical and experimental analysis show that the proposed algorithms are tight to a constant factor of existing algorithms for two primes, while increasing the generality of the cryptosystem.

The rest of this paper is structured as follows. Section II introduces related works on Chinese Remainder Theorem and the existing methods to solve the square root of quadratic residues modulo a prime p. Section III describes existing methods and algorithms necessary to develop the proposed theorems and algorithms. Section IV proposes a number of theorems, which can be applied to solving system of linear

The associate editor coordinating the review of this manuscript and approving it for publication was Rasheed Hussain.

congruences modulo n, and several algorithms based on the proposed theorems. Section V discusses the protocol to create ciphertext indistinguishability. Section VI conducts extensive experimental analysis by comparing the proposed algorithms with existing algorithms. The proposed theorems and algorithms extend the classical Rabin cryptosystem by allowing a public key to be the product of arbitrary finite number of distinct primes. Finally, Section VII concludes the paper and discusses future work.

II. RELATED WORK

In this section, some existing algorithms are presented to solve the quadratic residue problem modulo a prime p. An explanation of the Chinese Remainder Theorem and the equivalence to a single linear congruence, which is used to construct the proposed method to solve for quadratic residues modulo any natural number n.

Over the past 150 years, several algorithms have been developed to solve the quadratic residue problem modulo a single prime p. Tonelli in 1891 [9] proposed one of the first efficient algorithms which was later improved by Shanks in 1973 [10]. Lindhurst [11] showed that Tonelli's algorithm in $\mathbb{F}_p$ uses $O((\log p)^3)$ bit operations on average, when p and a are chosen at random. Cipolla developed the current most efficient algorithm which makes use of Field-Theoretic methods [12]. These algorithms are useful in situations where no condition on p is given aside from primality. Suppose a situation arises where $x^2 \equiv a \bmod p$. To make it a square on the right hand side, one may use either of these methods, for which the algorithms are described in Section IV. Faster algorithms for solving the quadratic residue problem exist if conditions are added to p. When $p \equiv 3 \pmod 4$, it is possible to apply Fermat's Little Theorem in order to solve the quadratic residue problem [13]. This algorithm has a time complexity of $O(log(b))$ which makes it popular for cryptographic applications.

The quadratic residue problem has applications to some forms of elliptic curve cryptography. Various protocols exist for this type of cryptosystem, which can implement encryption, decryption, and signing operations [14]. In general, these procedures are computed by evaluating points on an elliptic curve of the form $y^2 = x^3 + ax + b \pmod p$ for some chosen integers a and b which constitutes the Elliptic Curve Discrete Logarithm Problem (ECDLP) [14]. In elliptic curve cryptography, points on the curve are often stored using point compression [15]. In point compression, a point (x, y) on the elliptic curve can be stored as (x, β) where β is a single bit of y. Point decompression, that is, restoring the value of y from (x, β), requires computing the square root of a quadratic residue.

The Rabin cryptosystem is one of the most direct cryptographic applications of the quadratic residue problem. This protocol, published in January 1979 by Rabin, was the first asymmetric cryptosystem to prove that recovery of the entire plaintext from the ciphertext is as hard as factoring integers [8]. In the original method, Rabin introduced $x^2 \equiv c \pmod n$, where c is the ciphertext and $n = pq$ in which p and q are distinct large primes. One main problem of Rabin cryptosystem is that there are at most four solutions when solving the quadratic residue problem. Several schemes were developed to overcome this issue: build redundancy in the message, send extra bits, or limit the message size and factors for n [13].

Limiting the message size and the factors for n are discussed by Takagi [16], Asbullah and Ariffin [17], and Mahad *et al.* [18]. Takagi first proposed the scheme which takes $n = p^2q$, with p and q distinct large primes, with no restriction on the message size M. Asbullah and Ariffin showed a scheme which takes $n = p^2q$, for $p, q \equiv 3 \pmod 4$ distinct large primes and limits the message size M to be $M \in (0, 2^{2n-2})$. Both schemes need to utilize Algorithms 1, 2, or 3 if $p, q \not\equiv 3 \pmod 4$ or $p, q \not\equiv 5 \pmod 8$. If $p, q \equiv 3 \pmod 4$, then Fermat's Little Theorem can easily be applied to solve each congruence.

Other than applying the Chinese Remainder Theorem to the quadratic residue problem in cryptography, there are other uses, for example in communication, vehicular technology, signal processing, and optical security. In the field of communication, Chen and Lin discussed an energy-efficient Media Access Control (MAC) address scheme using the Chinese Remainder Theorem for wireless sensor networks [2]. In the field of vehicular technology, Wang *et al.*, used the Chinese Remainder Theorem ranging method based on dual-frequency measurements [4]. Li *et al.*, also used a similar method in their phase-detection-based range estimation [3]. In the field of signal processing, Wang and Xia used the Chinese Remainder Theorem to perform analysis on performance [6]. In [5], Wang *et al.* discussed the largest dynamic range of a generalized Chinese Remainder Theorem for two integers. In [19], the authors suggest a hybrid digital and optical system for the future of fast and secure cryptography. Optical computing can provide a level of efficiency not achievable by digital systems alone, but trade offs in data integrity during decryption create new problems unique to optical cryptosystems. One approach to optical cryptosystems is given as a triplet of functions by Yatish *et al.* which describes a single function on a plaintext input matrix to encrypt and two functions to decrypt the ciphertext [20].

The new algorithms proposed in Section IV are applied to the Rabin cryptosystem for its direct application of the quadratic residue problem. The algorithms in Section IV can use any of the below methods to compute the square root modulo a prime p. It is shown how to generate the solution to the square root of a quadratic residue modulo $n = p_1^{k_1}p_2^{k_2}\cdots p_r^{k_r}$, which has at most 2^r unique solutions. Classically, the Chinese Remainder Theorem is used to generate the solutions (x modulo n). The following section presents existing algorithms to solve the quadratic residue problem modulo a prime p and describes some existing results on the Chinese Remainder Theorem. In our experiment, we provide r less than 9 primes. When r is greater than 8 the computation time increases quickly with the current implementation. As a future work, we are working on a dynamic programming solution for the scenarios where r is greater than 8. The proposed algorithms in Section IV are developed from some extensions of these existing results and algorithms.

III. PRELIMINARIES

Tonelli published the first algorithm in 1893 [9], shown in Algorithm 1. This is one of the probabilistic algorithms to compute the quadratic residue modulo p. There is a probability of at least 1/2 for a random element of $\mathbb{F}_p$ to be a non-square integer. It takes input a from the congruence $x^2 \equiv a \pmod p$ and computes b where $b^2 = a$. The time complexity of Tonelli's algorithm is $O((logp)^4)$.

Algorithm 1 Tonelli's Algorithm

```
Input: $a \in \mathbb{F}_p^*$, $p$ odd
Output: $b = \sqrt{a}$ on $\mathbb{F}_p^*$
 1: Choose $g \in \mathbb{F}_p^*$ at random
 2: if $g$ is a square then
 3:     return -1
 4: end if
 5: Let $p - 1 = 2^s t$, $t$ odd.
 6: $e \leftarrow 0$.
 7: for $i \leftarrow 2$ to $s$ do
 8:     if $(ag^{-e})^{(p-1)/2^i} \neq 1$ then
 9:         $e \leftarrow 2^{i-1} + e$
10:     end if
11: end for
12: $h \leftarrow ag^{-e}$
13: $b \leftarrow g^{e/2} h^{(t+1)/2}$
14: return $b$
```

Cipolla published another algorithm in 1903 [12], shown in Algorithm 2. This is the current best known algorithm for calculating the square root of a quadratic residue modulo a prime p with no restriction on the choice of p. Cipolla's algorithm constructs a quadratic extension of $\mathbb{F}_p$, and uses an element t whose norm $N(t)$ equals a [21]. The time complexity of Cipolla's algorithm is $O((logp)^3)$.

Algorithm 2 Cipolla's Algorithm

```
Input: $a \in \mathbb{F}_p^*$
Output: $b = \sqrt{a}$ on $\mathbb{F}_p^*$
  Choose $t \in \mathbb{F}_p^*$ at random
  if $t^2 - 4a$ is a square then
      return -1
  end if
  $f \leftarrow X^2 - tX + a$
  $b \leftarrow X^{(p+1)/2} \pmod f$
  return $b$
```

The square root modulo p may be determined by using the definition of modulo, $b^2 = a + kn, \exists k, n \in \mathbb{N}, 1 \leq k \leq \lfloor \frac{n}{4} \rfloor$, as shown in Algorithm 3.

Algorithm 3 Algorithm by Exhaustion

```
Input: $a \in \mathbb{F}_p^*$
Output: $b = \sqrt{a}$ on $\mathbb{F}_p^*$
   $s \leftarrow a$
   while $s$ is not a square do
3:     $s \leftarrow s + a$
   end while
   $b \leftarrow \sqrt{s}$
6: return $b$
```

Since the speed of calculating the square root is of high interest in cryptographic applications and computing in general, some constraints may be added to the choice of a prime p where faster algorithms exist to compute the square root. The most popular algorithm utilized is the algorithm based on Fermat's Little Theorem [13]. The algorithm is given in Algorithm 4.

Algorithm 4 Algorithm Based on Fermat's Little Theorem

```
Input: $a \in \mathbb{F}_p^*$ and $p \equiv 3 \pmod 4$
Output: $b = \sqrt{a}$ on $\mathbb{F}_p^*$
  $b \leftarrow a^{\frac{p+1}{4}} \pmod p$
  return $b$
```

The first theorem presents the classical method for the Chinese Remainder Theorem.

Theorem 1 (Classical Chinese Remainder Theorem): Let $m_1, m_2, \ldots, m_r$ be pairwise relatively prime integers. Then the simultaneous congruence

$$\begin{cases} x \equiv a_1 \pmod{m_1} \\ x \equiv a_2 \pmod{m_2} \\ \quad \vdots \\ x \equiv a_r \pmod{m_r} \end{cases} \tag{1.1}$$

has a unique solution modulo the product $m_1 \times m_2 \times \cdots \times m_r$.

Proof: See Appendix A for proof of (1.1) [22]. □

The next theorem is proposed by Nagasaka *et al.* [23]. It is an improvement to the Chinese Remainder Theorem. This theorem will be the basis for our proposed theorems.

Theorem 2: (Improved Algorithm) Under the same assumptions as in the Chinese Remainder, the system of congruences (1.1) is equivalent to the following single linear congruence.

$$(\sum_{i=1}^{r} b_i M_i)x \equiv \sum_{i=1}^{r} a_i b_i M_i \pmod M$$

where b_i's are arbitrary integers coprime to m_i's, respectively, and $M_i = M/m_i$ for $i = 1, 2, \ldots, r$ with

$$M = \prod_{i=1}^{r} m_i$$

Proof: See Appendix B [23]. □

IV. PROPOSED THEORIES AND ALGORITHMS

Before presenting the proposed algorithms, two novel theorems and corresponding proofs on the Chinese Remainder Theorem are developed. The result is then extended to the quadratic residue problem. Finally, five generalized algorithms are presented based on the proposed theorems which can be utilized for decryption in the extended Rabin cryptosystem. The algorithms can be modified to solve point decompression in an elliptic curve cryptosystem.

Lemma 3: Let a system of linear congruences be given:

$$\begin{cases} x \equiv a_1 \pmod{m_1} \\ x \equiv a_2 \pmod{m_2} \\ \quad\vdots \\ x \equiv a_r \pmod{m_r} \end{cases}$$

where $\gcd(m_i, m_j) = 1$, *for* $1 \le i, j \le r, i \ne j$. *Suppose* $\sum_{i=1}^{n} b_i M_i = 1$, *where* $\gcd(b_i, m_i) = 1$ *and* $M_i = \frac{m_1 \times m_2 \times \cdots \times m_r}{m_i}$. *Then the system has a unique solution:*

$$x \equiv \sum_{i=1}^{n} a_i b_i M_i \pmod{M},$$

where $M = m_1 \times m_2 \times \cdots \times m_r$.

Proof: From Theorem 1, the system has a unique solution. From Theorem 2, the system is equivalent to

$$(\sum_{i=1}^{r} b_i M_i)x \equiv \sum_{i=1}^{r} a_i b_i M_i \pmod{M}$$

Since $\sum_{i=1}^{r} b_i M_i = 1$, the result follows. □

This result is extended to solve the quadratic residue problem. Theorem 4 solves the general version of the quadratic residue problem using the linear congruence from Lemma 3 for any n, where $1 \le i \le r, k_i \in \mathbb{N}, n = p_1^{k_1} p_2^{k_2} \cdots p_r^{k_r}$. Special cases of Lemma 3 which are valuable in cryptography are given as corollaries.

Theorem 4: Let $x^2 \equiv a^2 \pmod{n}$, *where* $n = p_1^{k_1} p_2^{k_2} \cdots p_r^{k_r}$, $p_1, \ldots, p_r$ *are distinct primes, and* $k_1, k_2, \ldots, k_r \in \mathbb{N}$. *If* $\sum_{i=1}^{r} b_i \frac{n}{p_i^{k_i}} = 1$, *and* $(b_1, p_1) = (b_2, p_2) = \cdots = (b_r, p_r) = 1$, *then there are* 2^r *solutions, which are:*

$$x \equiv a(\pm b_1 \frac{n}{p_1^{k_1}} \pm b_2 \frac{n}{p_2^{k_2}} \pm \cdots \pm b_r \frac{n}{p_r^{k_r}}) \pmod{n}$$

or equivalently, $x \equiv x_{j_1, j_2, \cdots, j_r} \equiv a[(-1)^{j_1}(b_1 \frac{n}{p_1^{k_1}}) + (-1)^{j_2}(b_2 \frac{n}{p_2^{k_2}}) + \cdots + (-1)^{j_r}(b_r \frac{n}{p_r^{k_r}})] \pmod{n}, j_l = 0, 1,$ *for* $1 \le l \le r$.

Proof: First, the square root on both sides is taken to obtain $x \equiv \pm a \pmod{n}$. Then there exists the following system of congruences:

$$\begin{cases} x \equiv \pm a \pmod{p_1^{k_1}} \\ x \equiv \pm a \pmod{p_2^{k_2}} \\ \quad\vdots \\ x \equiv \pm a \pmod{p_r^{k_r}} \end{cases}$$

Since $\sum_{i=1}^{r} b_i \frac{n}{p_i^{k_i}} = 1$, Lemma 3 is applied directly. Therefore, the solutions are:

$$x \equiv a(\pm b_1 \frac{n}{p_1^{k_1}} \pm b_2 \frac{n}{p_2^{k_2}} \pm \cdots \pm b_r \frac{n}{p_r^{k_r}}) \pmod{n}$$

or equivalently, $x \equiv x_{j_1, j_2, \cdots, j_r} \equiv a[(-1)^{j_1}(b_1 \frac{n}{p_1^{k_1}}) + (-1)^{j_2}(b_2 \frac{n}{p_2^{k_2}}) + \cdots + (-1)^{j_r}(b_r \frac{n}{p_r^{k_r}})] \pmod{n}, j_l = 0, 1,$ for $1 \le l \le r$. □

Following from Theorem 4 are two corollaries. Corollary 5 gives a formula for the solutions of a quadratic residue problem where $n = p_1^{k_1} p_2^{k_2}$, for which there are four solutions.

Corollary 5: Let $x^2 \equiv a^2 \pmod{n}$, *where* $n = p_1^{k_1} p_2^{k_2}$, p_1, p_2 *are distinct primes, and* $k_1, k_2 \in \mathbb{N}$.

If $b_1 \frac{n}{p_1^{k_1}} + b_2 \frac{n}{p_2^{k_2}} = 1$, *where* $(b_1, p_1) = (b_2, p_2) = 1$, *then*

$$x \equiv a(\pm b_1 \frac{n}{p_1^{k_1}} \pm b_2 \frac{n}{p_2^{k_2}}) \pmod{n}$$

Similarly, there exists a formula ($n = p_1^{k_1} p_2^{k_2} p_3^{k_3}$) for the solutions of quadratic residue problem for which there are eight solutions given by Corollary 6.

Corollary 6: Let $x^2 \equiv a^2 \pmod{n}$, *where* $n = p_1^{k_1} p_2^{k_2} p_3^{k_3}$, p_1, p_2, p_3 *are distinct primes, and* $k_1, k_2, k_3 \in \mathbb{N}$.

If $b_1 \frac{n}{p_1^{k_1}} + b_2 \frac{n}{p_2^{k_2}} + b_3 \frac{n}{p_3^{k_3}} = 1$, *where* $(b_1, p_1) = (b_2, p_2) = (b_3, p_3) = 1$, *then*

$$x \equiv a(\pm b_1 \frac{n}{p_1^{k_1}} \pm b_2 \frac{n}{p_2^{k_2}} \pm b_3 \frac{n}{p_3^{k_3}}) \pmod{n}$$

Given a situation in which the quadratic residue is not in the form of a perfect square, there exist several methods to solve $x^2 \equiv a \pmod{n}$ (1), assuming the congruence (1) is solvable. Lemma 3 requires a perfect square on the right hand side. However, this can be overcome. Using the prime factorization of $n = p_1^{k_1} p_2^{k_2} \cdots p_r^{k_r}$, Algorithms 1 and 2 can be applied to solve square roots in $\mathbb{F}_{p_i}$, $1 \le i \le r$, as discussed in Section III, then an application of Hensel's lemma [24] can lift each solution to $\mathbb{F}_{p_i^{k_i}}$. Alternatively, Algorithm 3 can be used to directly find the perfect square or subject to certain constraints discussed in Section III, Algorithm 4 can be utilized with high performance. Then, the following Theorem, Theorem 7, will obtain all of the 2^r solutions to (1).

Theorem 7: Let $x^2 \equiv a \pmod{n}$, *where* $n = p_1^{k_1} p_2^{k_2} \cdots p_r^{k_r}$, $p_1, \ldots, p_r$ *are distinct primes, and* $k_1, k_2, \ldots, k_i \in \mathbb{N}$.

If $\sum_{i=1}^{r} b_i \frac{n}{p_i^{k_i}} = 1$, *where* $(b_1, p_1) = (b_2, p_2) = \cdots = (b_r, p_r) = 1$, *then there are* 2^r *solutions, which are:*

$$x \equiv \pm a_1 b_1 \frac{n}{p_1^{k_1}} \pm a_2 b_2 \frac{n}{p_2^{k_2}} \pm \cdots \pm a_r b_r \frac{n}{p_r^{k_r}} \pmod{n}$$

where $a_i^2 \equiv a \pmod{p_i^{k_i}}$, *or equivalently:* $x \equiv x_{j_1, j_2, \cdots, j_r} \equiv (-1)^{j_1}(a_1 b_1 \frac{n}{p_1^{k_1}}) + (-1)^{j_2}(a_2 b_2 \frac{n}{p_2^{k_2}}) + \cdots + (-1)^{j_r}(a_r b_r \frac{n}{p_r^{k_r}}) \pmod{n}, j_l = 0, 1,$ *for* $1 \le l \le r$.

Proof: First reduce the congruence to the following:

$$\begin{cases} x^2 \equiv a \pmod{p_1^{k_1}} \\ x^2 \equiv a \pmod{p_2^{k_2}} \\ \quad\vdots \\ x^2 \equiv a \pmod{p_r^{k_r}} \end{cases}$$

After reducing modulo $p_i^{k_i}$ in each congruence, the system becomes:

$$\begin{cases} x^2 \equiv c_1 \pmod{p_1^{k_1}} \\ x^2 \equiv c_2 \pmod{p_2^{k_2}} \\ \quad\vdots \\ x^2 \equiv c_r \pmod{p_r^{k_r}} \end{cases}$$

Note that reducing modulo $p_i^{k_i}$ is not necessary. To obtain a square on the right hand side, any of the methods presented in Section III can be utilized. Let $a_i^2 \equiv c_i \pmod{p_i^{k_i}}$, then the system becomes:

$$\begin{cases} x^2 \equiv a_1^2 \pmod{p_1^{k_1}} \\ x^2 \equiv a_2^2 \pmod{p_2^{k_2}} \\ \quad\vdots \\ x^2 \equiv a_r^2 \pmod{p_r^{k_r}} \end{cases}$$

Taking the square root of both sides, the following system is obtained:

$$\begin{cases} x \equiv \pm a_1 \pmod{p_1^{k_1}} \\ x \equiv \pm a_2 \pmod{p_2^{k_2}} \\ \quad\vdots \\ x \equiv \pm a_r \pmod{p_r^{k_r}} \end{cases}$$

Combining Lemma 3 and Theorem 4, the solutions are:

$$x \equiv \pm a_1 b_1 \frac{n}{p_1^{k_1}} \pm a_2 b_2 \frac{n}{p_2^{k_2}} \pm \cdots \pm a_r b_r \frac{n}{p_r^{k_r}} \pmod{n}$$

or equivalently, $x \equiv x_{j_1,j_2,\cdots,j_r} \equiv (-1)^{j_1}(a_1 b_1 \frac{n}{p_1^{k_1}}) + (-1)^{j_2}(a_2 b_2 \frac{n}{p_2^{k_2}}) + \cdots + (-1)^{j_r}(a_r b_r \frac{n}{p_r^{k_r}}) \pmod{n}, j_l = 0, 1,$ for $1 \le l \le r$. □

The following new algorithm to compute the square root of a quadratic residue modulo $n = p_1^{k_1} p_2^{k_2} \cdots p_r^{k_r}$ is presented due to the results of Theorem 4. The first algorithm is used to compute the quadratic residue when $x^2 \equiv a^2 \pmod{n}$, where $n = p_1^{k_1} p_2^{k_2} \cdots p_r^{k_r}$. Given the perfect square associated with the quadratic residue $a^2 \pmod{n}$, which can be determined using Algorithm 4 in Section III, then the following algorithm enumerates the 2^r solutions by reducing the problem to a calculation of the extended Greatest Common Divisor (GCD). The extended GCD of two numbers a and b with $\gcd(a, b) = g$ is a solution $m, n \in \mathbb{Z}$ to the linear Diophantine equation $ma + nb = g$. The extended GCD of r integers, $w_1, w_2, \ldots, w_r$ is a solution $b_1, b_2, \ldots, b_r \in \mathbb{Z}$ that satisfies the linear Diophantine equation $b_1 w_1 + b_2 w_2 + \cdots + b_r w_r = g$. An algorithm is also presented to compute the extended GCD of r integers in Algorithm 8 of Section IV.

If only the coefficients are required, then the algorithm runs in $O(rQ(b) + G(b))$ time, where $Q(b)$ is the time complexity required to compute the square root of a quadratic residue of b bits given by Algorithms 1 - 4 in Section III. If Algorithm 4 is used, then $Q(b) = O(log(b))$. $G(b)$ is the time required to compute the Extended GCD of r numbers, for which the given algorithm runs in $O(rM(b)log(b))$. Therefore, the time complexity to generate the terms for the solutions is $O(rM(b)log(b))$. However, the time complexity to enumerate all solutions is $O(2^r)$. To determine the correct plaintext, padding bits can be used and checked during enumeration.

The output $\boldsymbol{x_{2^r}}$ are the 2^r solutions to the equation $x^2 \equiv a^2 \pmod{n}$ given by the vector $\boldsymbol{x_{2^r}} = \begin{pmatrix} x_0 & x_1 & \cdots & x_{2^r} \end{pmatrix}$. Algorithm 9 is presented to compute all of the 2^r solutions using recursion, which can be easily implemented.

Algorithm 5 can be further generalized for input of the perfect square associated with each quadratic residue $a_1^2 \pmod{p_1^{k_1}}$, $a_2^2 \pmod{p_2^{k_2}}$, $\ldots$, $a_r^2 \pmod{p_r^{k_r}}$, which can be calculated using Tonelli's algorithm [10], Cipolla's algorithm [12] or the algorithm by exhaustion with no added condition on the primes, or by using Fermat's Little Theorem [13] if the added condition $p_i \equiv 3 \pmod 4$ holds to calculate the perfect squares in mod $p_1, p_2, \ldots, p_r$ spaces. Hensel's lemma [24] can be applied to lift these solutions to the mod spaces $p_1^{k_1}, p_2^{k_2}, \ldots, p_r^{k_r}$.

Algorithm 5 Decryption Given $a^2 \pmod{n}$

```
Input: $a^2 \pmod{n}$ and $n = p_1^{k_1} p_2^{k_2} \cdots p_r^{k_r}$
Output: $\boldsymbol{x_{2^r}} = x_1, x_2, \cdots, x_{2^r}$ solutions to square root
    $a \leftarrow \pm\sqrt{a^2}$
    $b1, b2, \cdots, b_r \leftarrow$ ExtendedGCD$\left(\frac{n}{p_1^{k_1}}, \frac{n}{p_2^{k_2}}, \cdots, \frac{n}{p_r^{k_r}}\right)$
    for $j \leftarrow 1$ to $r$ do
        if $\gcd(b_j, p_j) \neq 1$ then
5:          return -1
        end if
    end for
    return $\boldsymbol{x_{2^r}}$
```

The output $\boldsymbol{x_{2^r}}$ are the 2^r solutions to the equation $x^2 \equiv a^2 \pmod{n}$.

A complete algorithm to find all 2^r solutions is given in Algorithm 7 that takes any quadratic residue a and r primes raised to arbitrary powers $p_1^{k_1}, p_2^{k_2}, \ldots, p_r^{k_r}$. Various choices for implementations of the PerfectSquare and GenSqrRt functions can be taken. However, for maximum speed, imposing $p \equiv 3 \pmod 4$ and using Algorithm 4 in Section III to implement PerfectSquare and using Algorithm 6 in Section IV to implement GenSqrRt gives Algorithm 7 and 9 together a $O(rM(b)log(b) + 2^r)$ time complexity when run together, where r is the number of primes. When $r = 2$, this algorithm has $O(M(b)log(b))$ time complexity which is asymptotically tight to the existing algorithm for the Rabin cryptosystem.

Algorithm 6 Generalized Decryption Algorithm

```
Input: $a_1^2 \pmod{p_1^{k_1}}, a_2^2 \pmod{p_2^{k_2}}, \ldots, a_r^2 \pmod{p_r^{k_r}}$ perfect
    squares of quadratic residues and $n = p_1^{k_1} p_2^{k_2} \cdots p_r^{k_r}$
Output: $\boldsymbol{x_{2^r}} = x_1, x_2, \cdots, x_{2^r}$ solutions to square root
    for $i \leftarrow 1$ to $r$ do
        $a_i \leftarrow \pm\sqrt{a_i^2}$
    end for
    $b1, b2, \cdots, b_r \leftarrow$ ExtendedGCD$\left(\frac{n}{p_1^{k_1}}, \frac{n}{p_2^{k_2}}, \cdots, \frac{n}{p_r^{k_r}}\right)$
    for $j \leftarrow 1$ to $r$ do
6:      if gcd$(b_j, p_j) \neq 1$ then
            return -1
        end if
    end for
    return $\boldsymbol{x_{2^r}}$
```

Algorithm 7 General Rabin Decryption Algorithm

```
Input: $a \in \mathbb{Z}_n$ and $n = p_1^{k_1} p_2^{k_2} \ldots p_r^{k_r}$
Output: $\boldsymbol{x_{2^r}} = x_1, x_2, \cdots, x_{2^r}$ solutions to square root
    for $i \leftarrow 1$ to $r$ do
        $a_i^2 \leftarrow$ PerfectSquare$(a, p_i^{k_i})$
    end for
    $\boldsymbol{x_{2^r}} \leftarrow$ GenSqrRt$(a_1^2, a_2^2, \ldots, a_r^2, p_1^{k_1}, p_2^{k_2}, \ldots, p_r^{k_r})$
    return $\boldsymbol{x_{2^r}}$
```

Using the extended GCD function for two integers included in the GMP Library which has $O(M(b)log(b))$ time complexity [25], where b is the number of bits in the numbers to multiply and $M(b)$ is the time it takes to multiply two numbers of b bits. The ExtendedGCD (Algorithm 8) for r terms is given by the calculation of $r - 1$ extended GCD calculations of two numbers, mpz_gcdext(a, b), in Algorithm 8. This gives a time complexity for Algorithm 8 as $O(rM(b)log(b))$. To solve the ExtendedGCD$(w_1, w_2, \ldots, w_r)$, Algorithm 8 solves the extended gcd of disjoint pairs of $w_1, w_2, \ldots, w_r$. Then the GCD g_{ij} of each pair, w_i, w_j is used as the values for the next recursive call of ExtendedGCD. In the first call of ExtendedGCD, $\frac{r}{2}$ extended GCD's of two numbers are computed and in the second call $\frac{r}{4}$ are computed. The algorithm proceeds in this manner until there is only one value left, which is the GCD of $w_1, w_2, \ldots, w_r$ and this is the stopping criterion for the function. Therefore, at level i, we have $\frac{r}{2^i}$ calls to mpz_gcdext. In sum, a geometric series is obtained: $\sum_{i=1}^{log_2(r)} \frac{r}{2^i} = r \sum_{i=1}^{log_2(r)} \frac{1}{2^i} = r$. Since each call is $O(M(b)log(b))$, then Algorithm 8 has $O(rM(b)log(b))$ time complexity.

In Algorithm 9, an implementation is given to enumerate all 2^r solutions. This algorithm can be used to find x_{2^r} in Algorithms 5 and 6 in $O(2^r)$ time complexity. The algorithm uses recursion to generate the solutions which builds the possible combinations for a coefficient of 1 or -1 for the sum of r terms from the possible combinations for the sum of $r - 1$ terms.

Algorithm 8 ExtendedGCD$(w_1, w_2, \ldots, w_r)$

```
Input: $b[]$ coefficients of extended GCD, $w[]$, $n$ size of array,
    gcd $g$
Output: $b_1, b_2, \cdots, b_i$ coefficients satisfying $b_1w_1+b_2w_2+\cdots+b_rw_r = 1$
    if $n < 2$ then
        $b[0] \leftarrow 1$
        return
    end if
    if $n \pmod 2 = 1$ then
        $s \leftarrow (n + 1)/2$
    else
8:      $s \leftarrow n/2$
    end if
    $k \leftarrow 0$
    for $i \leftarrow 0$ to $n - 2$ by 2 do
        mpz_gcdext$(neww[k], b[i], b[i + 1], w[i], w[i + 1])$
        $k \leftarrow k + 1$
    end for
    if $n \pmod 2 = 1$ then
16:     $neww[k] \leftarrow w[n - 1]$
        $b[n - 1] \leftarrow 1$
    end if
    if $n = 2$ then
        $g \leftarrow neww[k - 1]$
    end if
    ExtendedGCD$(g, newb, neww, s)$
    for $i \leftarrow 0$ to $s - 1$ do
24:     $b[2i] \leftarrow b[2i]newb[i]$
        $b[2i + 1] \leftarrow b[2i + 1]newb[i]$
    end for
    if $n \pmod 2 = 1$ then
        $b[n - 1] \leftarrow newb[s - 1]$
    else
        $b[2(s - 1)] \leftarrow b[2(s - 1)]newb[s - 1]$
        $b[2(s - 1) + 1] \leftarrow b[2(s - 1) + 1]newb[s - 1]$
32: end if
```

V. SECURITY ANALYSIS

In this section, the security aspect of the proposed algorithm is discussed. The Rabin cryptosystem relies on the difficulty of integer factorization; however, only two distinct large primes are used. The proposed algorithm allows for any fixed number of primes to be used, and the number can be changed between sessions allowing for increased unpredictability. The classical Rabin cryptosystem does not provide ciphertext indistinguishability. This is because the encryption process is deterministic. That is, given a public key n, to encrypt a message m the process follows $m^2 \equiv c \pmod n$. If an eavesdropper knows n and intercepts an encrypted message c, then to determine if c encodes a potential plaintext message $\hat{m}$, then the eavesdropper can perform $\hat{m}^2 \equiv c_1 \pmod n$. If $c_1 = c$, then $m = \hat{m}$ and the plaintext has been recovered. Formally stated, given a message space $m \in \{a, b\}$

Algorithm 9 GenerateSolutions

```
Input: terms of the solutions $a[]$, size of array $r$
Output: $x[]$ array of $2^r$ solutions
    if $r = 1$ then
        $x[0] \leftarrow a[0]$
        $x[1] \leftarrow a[1]$
        return $x$
    end if
    $r_y \leftarrow 1 \ll (r - 1)$
    $y \leftarrow$ GenerateSolutions($a, r - 1$)
    for $i \leftarrow 0$ to $r_y$ do
9:      $x[i] \leftarrow y[i] + a[r_y - 1]$
    end for
    for $i \leftarrow r_y$ to $2 * r_y$ do
        $x[i] \leftarrow -x[i - r_y]$
    end for
    return $x$
```

and a ciphertext c such that $E(m) = c$, a ciphertext gives indistinguishability if an adversary cannot determine which $m \in \{a, b\}$ encodes c with probabilty greater than $\frac{1}{2}$. In the classical Rabin system, an adversary can determine which plaintext m encodes ciphertext c by encrypting both candidates and choosing the correct plaintext by comparing with the captured ciphertext.

In the extended Rabin cryptosystem, ciphertext indistinguishability can be established. To do this, a random number, S can be added to the public key and a predetermined number, z, of least significant bits of the random number, S^z are used as the padding for the message. This creates a message space such that the adversary knowing $m \in \{a, b\}$ whereas the message is really in $m \in \{aS^z, bS^z\}$. Since the adversary does not know S, then they cannot determine a candidate message from only $\{a, b\}$. The public key can be extended using two large primes, p and q, and a random number S such that $n = pqS$. To generate S, each participant in the encryption agrees on a shared secret seed, s_0 and a pseudo random number generator f. Then, each message is sent using a new key derived from p, q, and s_0. The first message can be encrypted with $S = s_1 = f(s_0)$ and $n_1 = pqs_1$, and the i^{th} message can be encrypted with $s_i = f(s_{i-1})$ and $n_i = pqs_i$. In this way, the keys $n_1, n_2, \ldots$ do not have to be shared for subsequent encryptions. Therefore, an outside eavesdropper who intercepts the i^{th} ciphertext, c_i does not know $n_i = pqs_i$. Since s_i is unknown then the z least significant bits s_i^z are also unknown. Therefore, the eavesdropper cannot determine what padding to append to the candidate messages. However, since the two intended parties both know the initial seed s_0 and the function f, both intended parties know n_i for each message since the sequence $s_1, s_2, \ldots$ is deterministic given s_0. The observation that any given random number s_i has a prime factorization, $s_i = p_{s_i1}^{k_{s_i1}} p_{s_i2}^{k_{s_i2}} \cdots p_{s_ir}^{k_{s_ir}}$, then $n_i = pqs_i = pqp_{s_i1}^{k_{s_i1}} p_{s_i2}^{k_{s_i2}} \cdots p_{s_ir}^{k_{s_ir}}$. Therefore, the proposed algorithms in Section IV can be used to decrypt a ciphertext c_i with key n_i.

Given a public key $n = pq$ and shared secret (s_0, f), and a message space $m \in \{a, b\}$. Alice sends Bob an encrypted message c_i by computing $s_i = f(s_{i-1})$, constructing the key $n_i = pqs_i$, and encrypts the plaintext m_i with appended padding bits s_i^z using $m_i^2 \equiv c_i \pmod{n_i}$. Bob computes s_i and n_i independently of Alice using the same scheme, so n_i is not sent across the public channel. Then an adversary who knows n and intercepts a ciphertext corresponding to the i^{th} encryption, c_i, cannot determine m_i by encrypting candidate messages $\hat{m}$ by $\hat{m} \equiv \hat{c} \pmod{n}$ and comparing $\hat{c}$ to c_i because c_i was not encrypted using the public key n and the message m was appended with padding bits chosen from s_i. Since the choice of s_i is random, it is infeasible for the adversary to guess the key n_i even if n is known. It is similarly infeasible for the adversary to determine the padding used s_i^z for the message m_i since as long as z is greater than one, there are more than two choices for s_i^z.

VI. EXPERIMENTAL ANALYSIS

In this section, experimental evaluation is presented for the algorithms given in Section IV. The proposed algorithms are applied to the Rabin cryptosystem for its direct correspondence to the quadratic residue problem. For the experiments herein using Algorithm 7, the PerfectSquare function is implemented using Fermat's Little Theorem, Algorithm 4 in Section III. The GenSqrRt function is implemented using Algorithm 6 from Section IV. Three experiments are conducted. In the first experiment, $n = pq$ is fixed for a direct comparison with the proposed algorithm and the existing algorithm used to decrypt ciphertexts in the Rabin cryptosystem. The existing algorithm is derived from the classical Chinese Remainder Theorem (Theorem 1). The second experiment uses two primes and varies the number of bits in the prime to compare the runtime of our algorithm with the existing algorithm used to decrypt ciphertexts in the Rabin cryptosystem. Finally, the third experiment shows the run time of Algorithm 6 from Section IV as a function of the number of primes in the prime factorization of n.

A. SPECIFICATIONS

All of the algorithms for our experiments are implemented in C++11 using the g++ compiler version 7.3.0. For all of the tests, the −O3 flag is used for compiler optimizations. Due to the size of the integers in the calculations, the GNU MP Bignum Library (GMP) is used for integers with arbitrary precision [25]. To capture the run time of each test, the C++11 Chrono library's high resolution clock is used. The experiments were run in a 64-bit Ubuntu 18.04 LTS virtual machine on a Windows 10.0.17134 host operating system using VirtualBox 6.0.4 r128413 (Qt5.6.2). The virtual machine is allocated 2 hardware cores (4 logical) and 40,000 MB of RAM with VT-x/AMD-V, nested paging, and KVM paravirtualization. The host computer used to run the tests consists of a Intel Xeon E5-1630 v4 running at 3.70 GHz with 4 physical cores and 8 logical processors and 64 GB of RAM.

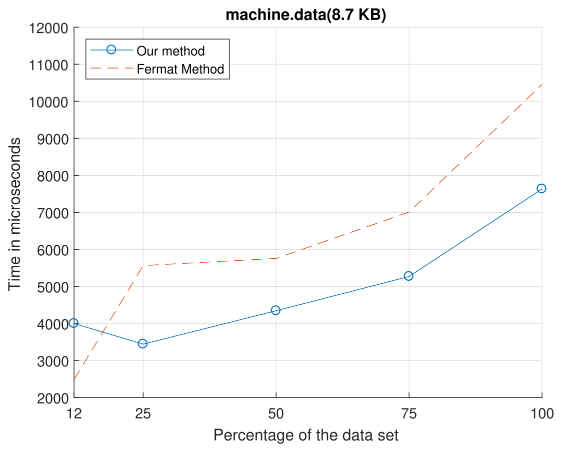

Figure 1a

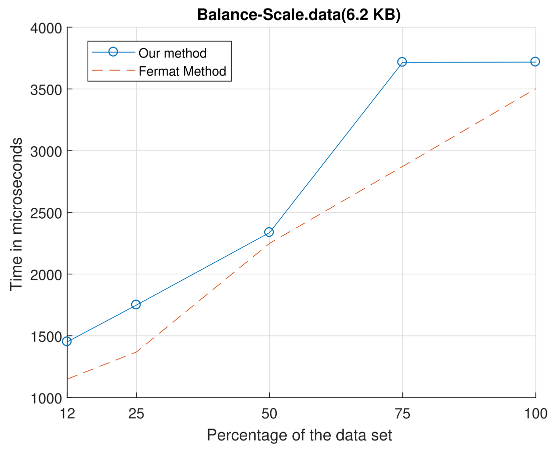

Figure 1b

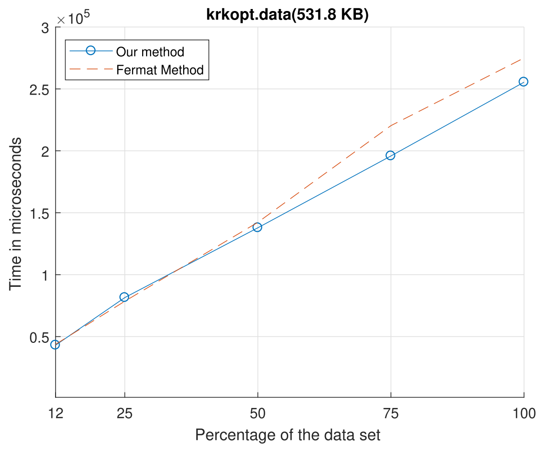

Figure 1c

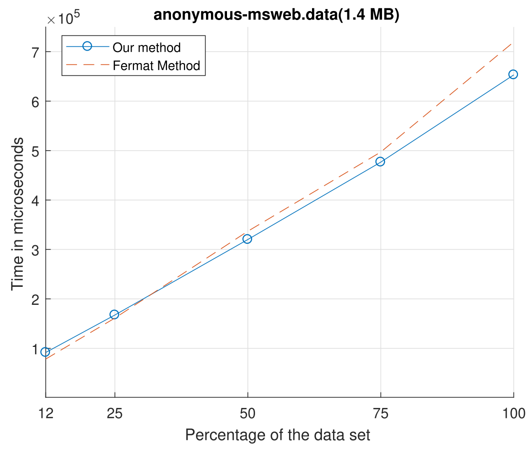

Figure 1d

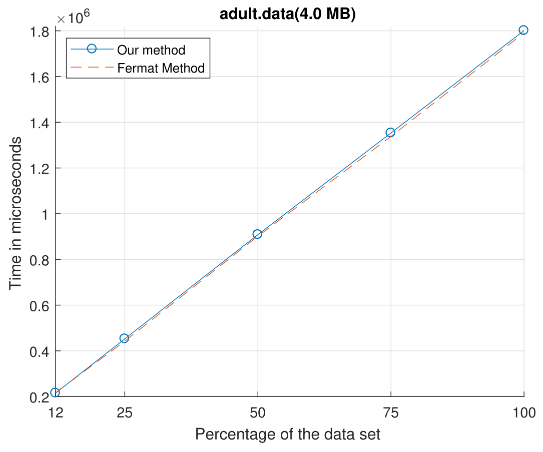

Figure 1e

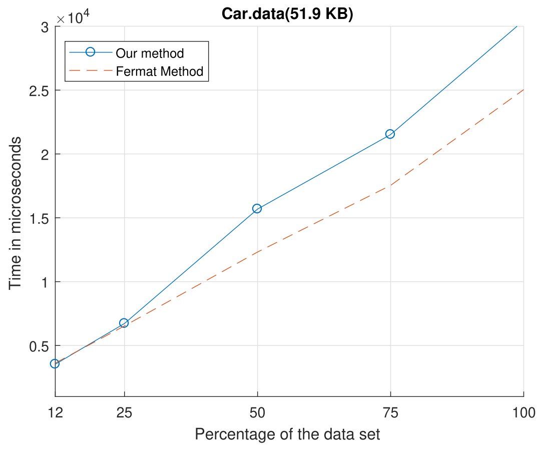

Figure 1f

FIGURE 1. Average run-time of 10 trials with varying size of plaintext shows the proposed method is competitive to the existing method when $n = pq$.

B. DATA DESCRIPTION

The six data sets used for the verification of the proposed method were obtained from UCI Machine Learning Repository [26]. The size of the data sets ranges from 6.2 KB to 4.0 MB. The different sizes of the data sets are chosen to demonstrate the performance of the algorithm under different circumstances. For example, one could apply the proposed algorithm to encrypt symmetric keys. The type and content of the data sets are not important to the encryption-decryption process.

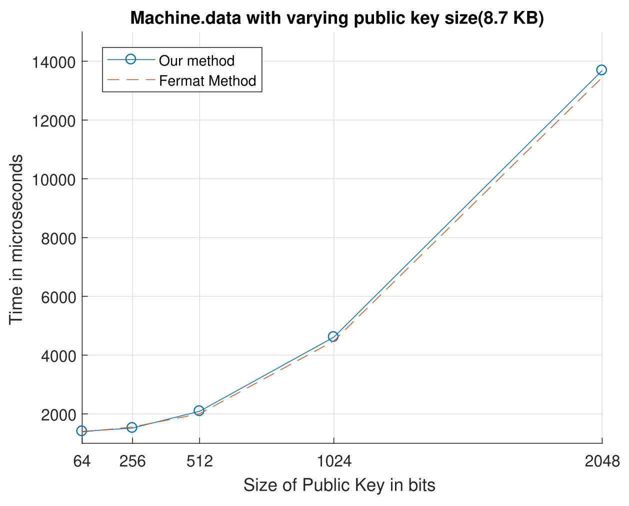

Figure 2a

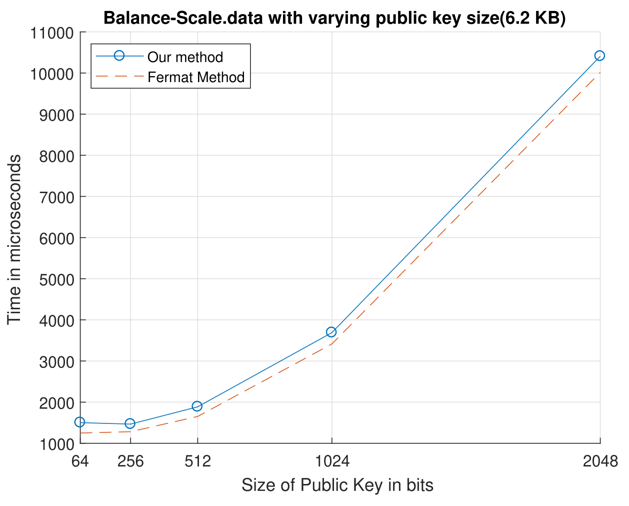

Figure 2b

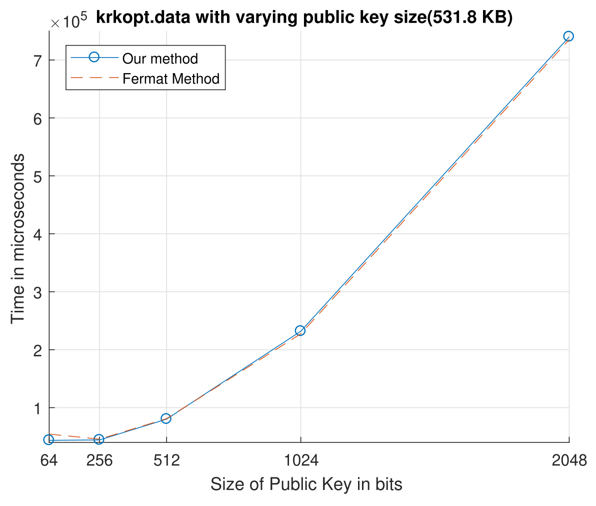

Figure 2c

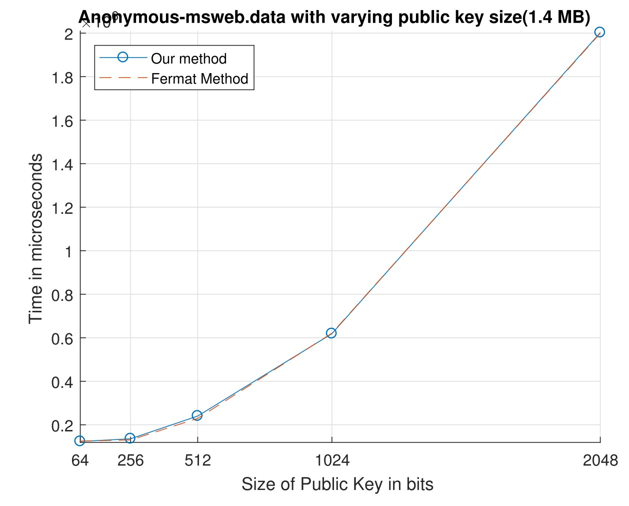

Figure 2d

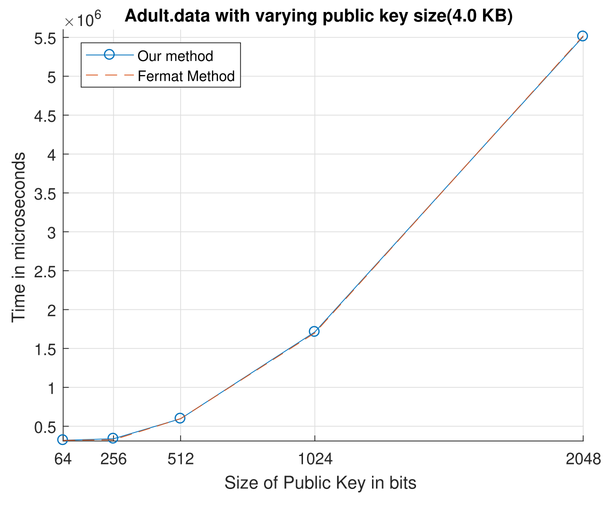

Figure 2e

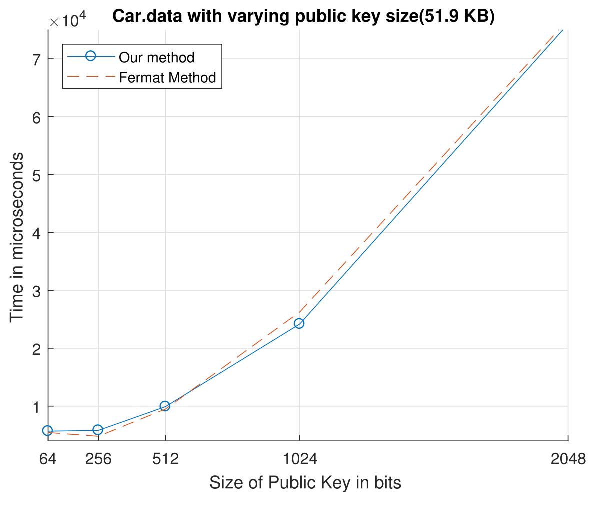

Figure 2f

FIGURE 2. Average run time of 10 trials with varying key size shows the proposed method is competitive when $n = pq$.

The Balance Scale data (balance-scale) set is the smallest data set that is tested in our experiments containing 6250 bytes of character encoded data. It consists of 625 instances of four attributes used to classify a scale as balanced, tipped to the right, or tipped to the left. The Computer Hardware data set (machine) contains 8726 bytes

TABLE 1. Performance comparison in microseconds highlights the superiority of the proposed method.

Data Set	Percentage of the data set	12	25	50	75	100
Machine	Fermat Method (μs)	2484.7	5562.4	5755.5	7009.1	10456
	Proposed Method (μs)	3997.7	**3441.6**	**4342.3**	**5268.5**	**7631**
krkopt	Fermat Method (μs)	43758	78560.7	142292.5	220350.7	274980.9
	Proposed Method (μs)	**43070**	81497.5	**137878.6**	**195869.9**	**255420.6**
Car	Fermat Method (μs)	3624	6534	12320	17540	25037
	Proposed Method (μs)	**3532**	6721	15682	21519	30448
Balance_scale	Fermat Method (μs)	1451	1749	2336	3715	37177
	Proposed Method (μs)	**1149**	**1369**	**2249**	**2872**	**3502**
Anonymous-msweb	Fermat Method (μs)	78375	161216	336841	497071	719737
	Proposed Method (μs)	91606	167405	**320115**	**476634**	**653084**
Adult	Fermat Method (μs)	217150	442520	901445	1338043	1791286
	Proposed Method (μs)	**215258**	452492	908394	1353018	1801056

of character encoded data. It consists of 209 instances of nine attributes to describe the relative performance of CPU chips. The Car Evaluation data (car) set contains 51867 bytes of character encoded data. It consists of 1728 instances of 6 attributes to evaluate various models of cars. The Chess Data Set (krkopt) consists of 531,806 character encoded bytes. The data set consists of 28,056 instances of 6 classifying attributes to evaluate common chess endgame position consisting of a white king and rook against a lone black king. The Anonymous Microsoft Web Data data set (anonymous-msweb) is a large data set of 1,423,098 character encoded bytes. The data set is 37,711 instances of 294 attributes compiled from user visits to the www.microsoft.com website. Finally, the Adult data set (adult) is the largest data set that is tested containing 3,974,305 character bytes. The data set is 48,842 instances of 14 attributes to classify the income brackets from census information. The data sets were chosen for their varying sizes, accessibility, and ease of use, rather than their application to machine learning. Each data set contains several character strings that are repeated. However, blocks of the plaintext relative to the key size are encrypted. Since the block size is larger than a single word, it is unlikely for two blocks to be exactly the same in the experiments.

C. EXPERIMENTAL RESULTS

The experiments block the large data sets into chunks. The block size is proportional to the key size and is small enough to avoid degenerate cases during encryption. The data sets were chosen in order to show clear trends in the algorithm's performance, rather than the application to encrypting symmetric keys. A degenerate case during encryption is one where two or more plaintexts map to the same cipher text. Both algorithms used the same encryption process.

Figures 1a-1f correspond to the first experiment and plot the average run-time in microseconds of each algorithm to the percentage of the data set encrypted. The data sets show that the proposed algorithm grows linearly with respect to the size of the plaintext. Fig. 1a shows the proposed algorithm performs better than the Fermat's method by about 28%. Fig. 1e shows the proposed algorithm has about the same run-time as Fermat's method. Fig. 1f is the only trial where the proposed algorithm did not perform better than the existing method. The data set used in Fig. 1f is relatively small, so the overhead associated with the proposed algorithm may have contributed to the slower performance. However, in larger data sets, this overhead is minimal. The results are displayed in Table 1 with the superior results shown in bold for increased clarity. It is shown that the proposed algorithm can provide a similar speed across a variety of data set sizes to the existing algorithm, while expanding the cryptosystem to allow an unlimited number of primes.

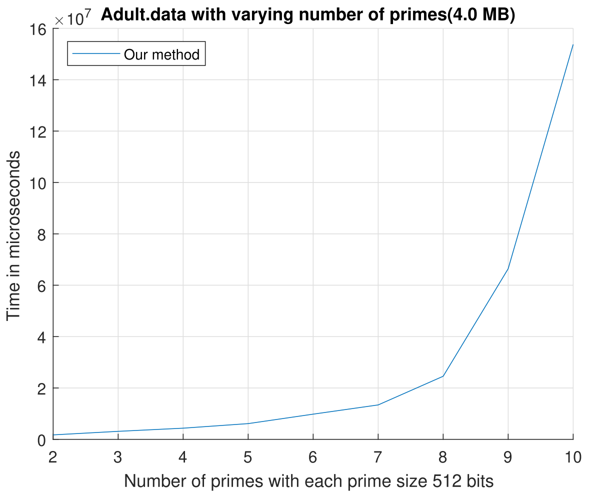

Figure 3a

FIGURE 3. The average run time is not greatly affected with less than 9 primes in the factorization of *n*.

Figures 2a-2f correspond to the second experiment, where the proposed algorithm is tested against Fermat's method with varying public key size. The experiment shows that the proposed algorithm has similar time complexity to the existing two prime algorithm for the Rabin cryptosystem independent of the key size. The experimental results for the proposed algorithm are consistent with the existing methods.

Figure 3a corresponds to the final experiment, where the proposed algorithm is tested with the public key composed of different number of primes. It is demonstrated in this experiment that the Rabin cryptosystem can be extended to provide an unlimited number of primes to construct the shared public key. In the experiment, the size of each prime is 512 bits. Figure 3a demonstrates that using eight or less primes in the prime factorization of the public key yields a negligible trade off in performance. Therefore, the proposed algorithm for indistinguishability given in Section V can be constructed with a random number s of no more than 4096 bits from 8 primes with 512 bits per prime, which represents a number in the range of $[0, 2^{4096}-1]$. Therefore, this approach maintains the efficiency of the unmodified extended Rabin cryptosystem. The graph is exponential in the number of primes due to the algorithm used to generate the 2^r solutions to the quadratic residue problem with r primes. The proposed algorithm used to enumerate the solutions runs in $O(2^r)$ time; however, an algorithm with polynomial time complexity can be implemented using a dynamic programming technique.

VII. CONCLUSION

This paper proposed a number of theorems and algorithms for the Chinese Remainder Theorem and the extended Rabin cryptosystem. The experimental analysis of the proposed algorithms being applied to Rabin cryptosystem shows a performance improvement compared with the existing algorithm for Rabin cryptosystem. The proposed algorithms also add support for encryption on any ring of integers modulo n with an arbitrary number of distinct primes in its prime factorization while retaining the elegance of a solution using only two primes. It was shown how this increase in generality can be used to provide ciphertext indistinguishability to the extended Rabin cryptosystem. Furthermore, by increasing the key size, a larger message space can be constructed to increase the size of the plaintext blocks. This can be used to reduce the fragmentation of large plaintexts which incur an increased number of decryptions in smaller message spaces. The algorithm to enumerate all 2^r solutions with polynomial time complexity is left as future work.

APPENDIXES

APPENDIX A

This proof is in Kumanduri and Romero's book [22].

Theorem 1 (Classical Chinese Remainder Theorem): Let $m_1, m_2, \ldots, m_r$ be pairwise relatively prime integers. Then the simultaneous congruence

$$\begin{cases} x \equiv a_1 \pmod{m_1} \\ x \equiv a_2 \pmod{m_2} \\ \quad\vdots \\ x \equiv a_r \pmod{m_r} \end{cases} \tag{1.1}$$

has a unique solution modulo the product $M = m_1 \times m_2 \times \cdots \times m_r$. *Proof:* Let $M_i = M/m_i$. Since the m_i's are pairwise relatively prime, m_i, so it has an inverse x_i modulo m_i; that is, $M_i x_i \equiv 1 \pmod{m_i}$. If $i \neq j$, then $m_i \mid M_j$; that is, $M_j \equiv 0 \pmod{m_i}$. Then

$$x \equiv a_1\, M_1\, x_1 + a_2\, M_2\, x_2 + \cdots + a_r M_r x_r \pmod{M}$$

is a solution to the system because

$$\begin{aligned} a_1\, M_1\, x_1 + a_2\, M_2\, x_2 + \cdots + a_r M_r x_r &\equiv a_i M_i x_i \pmod{m_i} \\ &\equiv a_i \cdot 1 \pmod{m_i} \\ &\equiv a_i \pmod{m_i} \end{aligned}$$

This proves the existence. For the uniqueness property of the solution:

Suppose x and y are two distinct solutions. Now $x \equiv y \pmod{m_i}$ for each i; that is, $m_i \mid (x-y)$. The m_i are pairwise relatively prime; hence, by unique factorization, the product $m_1 m_2 \cdots m_r \mid (x-y)$; that is, $x \equiv y \pmod{m_1 \cdots m_r}$. This proves the uniqueness. □

APPENDIX B

This proof is in Nagasaka *et al.*'s paper [23]. **Theorem 2** *(Improved Algorithm) Under the same assumptions as in the Chinese Remainder Theorem (Theorem 1), the system of congruences (1.1) is equivalent to the following single linear congruence.*

$$(\sum_{i=1}^{k} b_i M_i) x \equiv \sum_{i=1}^{k} a_i b_i M_i \pmod{M} \tag{1.2}$$

where b_i's are arbitrary integers coprime to m_i's, respectively, and $M_i = M/m_i$ for $i = 1, 2, \ldots, k$ with

$$M = \prod_{i=1}^{k} m_i$$

To prove this theorem, the following two lemmas are required.

Lemma 1: The system of congruences (1.1) is equivalent to the following system of congruences:

$$\begin{cases} M_1 x & \equiv a_1 M_1 \pmod{M} \\ M_2 x & \equiv a_2 M_2 \pmod{M} \\ & \quad\vdots \\ M_{k-1} x & \equiv a_{k-1} M_{k-1} \pmod{M} \\ \sum_{i=1}^{k} b_i M_i x & \equiv \sum_{i=1}^{k} a_i b_i M_i \pmod{M}, \end{cases} \tag{1.3}$$

where b_i is relatively prime to m_i for $i = 1, 2, \cdots, r$.

Proof: The necessity is shown as follows. Suppose x is a solution of (1.1). Then, for $i = 1, 2, \cdots, k$,

$$x - a_i = c_i m_i,$$

with an integer c_i. Multiplying the above identity with M_i for each i, we get for $i = 1, \cdots, k$,

$$M_i x - M_i a_i = c_i m_i M_i = c_i M,$$

which is rewritten as the congruence

$$M_i x \equiv M_i a_i \pmod{M} \tag{1.4}$$

for $i = 1, 2, \cdots, k$. The first $k-1$ congruences in (1.3) are identical with those of (1.4) for $i = 1, 2, \cdots, k-1$. The last congruence of (1.3) can be obtained as a linear combination of (1.4).

Conversely, we assume that x is a solution of the system of congruences. Since M_i and m_i are coprime, (1.4) can be reduced to

$$x \equiv a_i \pmod{m_i},$$

for $i = 1, 2, \cdots, k-1$.

Subtracting the linear combination of the first $k-1$ congruences in (1.3), we have

$$b_k M_k x \equiv a_k b_k M_k \pmod{M}.$$

Since M_k and b_k are relatively prime to m_k, we conclude that

$$x \equiv a_k \pmod{m_k}$$

This concludes the proof of Lemma 1.

Lemma 2: Under the same assumption as in Theorem III.2, M is relatively prime to $\sum_{i=1}^{k} b_i M_i$.

Proof. Let g be the GCD of $\sum_{i=1}^{k} b_i M_i$ and M and let p be a prime factor of g. Then p divides $\sum_{i=1}^{k} b_i M_i$ and $M = \sum_{i=1}^{k} m_i$. The divisibility of M by p implies that p divides only one of m_i, say m_j, since $\gcd(m_i, m_j) = 1$ if $i \neq j$. Then p divides all M_i but M_j, since $M_j = M/m_j$. Together with the divisibility of $\sum_{i=1}^{k} b_i M_i$ by p, we derive that p divides $b_j M_j$ and consequently divides b_j. Hence p divides the GCD of m_j and b_j which is equal to one from the assumption.

This concludes the proof of Lemma 2.

Proof of Theorem 1: By Lemma 1, we know that the system of congruences (1.1) is equivalent to the system (1.3). From Lemma 2, the last congruence has a unique solution, say $x_0 \pmod{M}$. Thus the system (1.3) has a unique solution x_0, which implies that x_0 is the unique solution of the system of congruences (1.1), since the Chinese Remainder Theorem assures the unique existence of the solution of the system (1.1) $\pmod{M}$.

ACKNOWLEDGMENT

The authors wish to express their sincere thanks to the referees for helpful comments and remarks that led to an improved version of the original manuscript.

REFERENCES

[1] P. Dingyi, S. Arto, and D. Cunsheng, *Chinese Remainder Theorem: Applications in Computing, Coding, Cryptography*. Singapore: World Scientific, 1996.

[2] Y.-S. Chen and Y.-W. Lin, "C-MAC: An energy-efficient mac scheme using chinese-remainder-theorem for wireless sensor networks," in *Proc. IEEE Int. Conf. Commun.*, Jun. 2007, pp. 3576–3581.

[3] X. Li, W. Wang, W. Zhang, and Y. Cao, "Phase-detection-based range estimation with robust Chinese remainder theorem," *IEEE Trans. Veh. Technol.*, vol. 65, no. 12, pp. 10132–10137, Dec. 2016.

[4] C. Wang, Q. Yin, and H. Chen, "Robust Chinese remainder theorem ranging method based on dual-frequency measurements," *IEEE Trans. Veh. Technol.*, vol. 60, no. 8, pp. 4094–4099, Oct. 2011.

[5] W. Wang, X. Li, X.-G. Xia, and W. Wang, "The largest dynamic range of a generalized Chinese remainder theorem for two integers," *IEEE Signal Process. Lett.*, vol. 22, no. 2, pp. 254–258, Feb. 2015.

[6] W. Wang and X.-G. Xia, "A closed-form robust Chinese remainder theorem and its performance analysis," *IEEE Trans. Signal Process.*, vol. 58, no. 11, pp. 5655–5666, Nov. 2010.

[7] R. A. Mollin, *An Introduction to Cryptography*. Boca Raton, FL, USA: CRC Press, 2006.

[8] M. O. Rabin, "Digitalized signatures and public-key functions as intractable as factorization," Massachusetts Inst. Tech. Cambridge Lab. Comput. Sci., Cambridge, MA, USA, Tech. Rep. ADA078415, 1979.

[9] A. Tonelli, "Bemerkung über die auflösung quadratischer congruenzen," *Nachrichten von der Königl. Gesellschaft der Wissenschaften und der Georg-Augusts-Universität zu Göttingen*. Göttingen, Germany: Gesellschaft der Wissenschaften, 1891, pp. 344–346.

[10] D. Shanks, "Five number-theoretic algorithms," in *Proc. 2nd Manitoba Conf. Numer. Math. (Winnipeg)*, 1973, pp. 51–70.

[11] S. Lindhurst, "An analysis of Shanks's algorithm for computing square roots in finite fields," in *Number Theory* (Lecture Notes), vol. 19. Montréal, QC, Canada: AMS and Centre de Recherches Mathématiques, 1999, pp. 231–242.

[12] M. Cipolla, "Un metodo per la risolutione della congruenza di secondo grado," *Rendiconto dell'Accademia delle Scienze Fisiche e Matematiche*, vol. 9. Naples, Italy: Napoli, pp. 154–163, 1903.

[13] J. Katz, A. J. Menezes, P. C. Van Oorschot, and S. A. Vanstone, *Handbook of Applied Cryptography*. Boca Raton, FL, USA: CRC Press, 1996.

[14] J. Lopez and R. Dahab, "An overview of elliptic curve cryptography," Inst. Comput., State Univ. Campinas, São Paulo, Brazil, Tech. Rep. IC-00-10, 2000.

[15] P. S. L. M. Barreto and J. F. Voloch, "Efficient computation of roots in finite fields," *Des., Codes Cryptogr.*, vol. 39, no. 2, pp. 275–280, May 2006.

[16] T. Takagi, "Fast RSA-type cryptosystems using n-adic expansion," in *Proc. Annu. Int. Cryptol. Conf.* Berlin, Germany: Springer, 1997, pp. 372–384.

[17] M. A. Asbullah and M. R. K. Ariffin, "Design of rabin-like cryptosystem without decryption failure," *Malaysian J. Math. Sci.*, vol. 10, pp. 1–18, Aug. 2016.

[18] Z. Mahad, M. Asbullah, and M. Ariffin, "Efficient methods to overcome rabin cryptosystem decryption failure," *Malaysian J. Math. Sci.*, vol. 11, pp. 9–20, Apr. 2017.

[19] B. Javidi, A. Carnicer, M. Yamaguchi, T. Nomura, E. Pérez-Cabré, M. S. Millán, N. K. Nishchal, R. Torroba, J. F. Barrera, and W. He, "Roadmap on optical security," *J. Opt.*, vol. 18, no. 8, 2016, Art. no. 083001.

[20] A. F. Yatish and N. K. Nishchal, "Optical image encryption using triplet of functions," *Opt. Eng.*, vol. 57, no. 3, 2018, Art. no. 033103.

[21] E. Bach, J. O. Shallit, J. Shallit, and S. Jeffrey, *Algorithmic Number Theory: Efficient Algorithms*, vol. 1. Cambridge, MA, USA: MIT Press, 1996.

[22] R. Kumanduri and C. Romero, *Number Theory With Computer Applications*. London, U.K.: Pearson, 1998.

[23] K. Nagasaka, J.-S. Shiue, and C.-W. Ho, "A fast algorithm of the Chinese remainder theorem and its application to Fibonacci numbers," in *Applications of Fibonacci Numbers*, vol. 4. Springer, 1991, pp. 241–246.

[24] K. H. Rosen, *Elementary Number Theory*. London, U.K.: Pearson, 2011.

[25] T. Granlund, *GNU MP 6.0 Multiple Precision Arithmetic Library*. Samurai Media Limited, 2015.

[26] A. Frank. (2010). *UCI Machine Learning Repository*. [Online]. Available: http://archive.ics.uci.edu/ml