Journal of Mathematics and Statistics 2 (1): 354-359, 2006
ISSN 1549-3644
© 2006 Science Publications

A Comparative Study of RSA Based Digital Signature Algorithms

[1]Ramzi A. Haraty, [2]A. N. El-Kassar and [1]Bilal Shibaro
[1]Lebanese American University P.O. Box 13-5053 Chouran, Beirut, Lebanon 1102 2801
[2]Beirut Arab University, Mathematics Department, Beirut, Lebanon

Abstract: A digital signature is a mechanism designed to allow secure communication through an insecure medium and can be traced in many applications where privacy is required. A digital signature is an electronic signature that can be used to authenticate the identity of the sender of a message or the signer of a document and possibly to ensure that the original content of the message or document that has been sent is unchanged. The main purpose of this study was to extend important and useful digital signature schemes from the domain of natural integers Z to two principal ideal domains; namely, the domain of Gaussian integers Z[i] and the domain of the ring of polynomials over finite fields F[x] by extending arithmetic needed for our extensions to these domains. We implement the classical and modified RSA cryptosystem to compare and to test their functionality, reliability and security. To test the security of the algorithms we implement attack algorithms to solve the factorization problem in Z, Z[*i*] and *F*[*x*]. After factorization is found, the RSA problem could be solved by finding the private key using the extended Euclidean algorithm.

Key words: Digital signatures, cryptosystem algorithms, testing and evaluation

INTRODUCTION

Digital signatures are strong tools applied in order to achieve the security services of authentication (proof of identity of the sender), data integrity (detection of changes to the message) and non-repudiation (prevention of denial of sending the information). They are digital counterpart of handwritten signatures that can be transmitted over a computer network. Only the sender can make the signature, but other people can easily recognize as belonging to the sender. The sender produces a signature consisting of a number associating a message (in digital form) with a secret key. This signature is intended to be unique and it does not necessarily require that a message be encrypted but must be verifiable. Diffie and Hellman introduced the concept of a digital signature in 1976. They published their landmark study "New Directions in Cryptography"[1]. Digital signatures schemes are based on one-way functions that are relatively easy to compute in one direction, but very difficult to compute going the other direction[2]. The RSA signature is the first method scheme discovered and is widely used[3]. The signature works in Z_n where n is the product of two large primes p and q and its security is based on the intractability of the integer factorization problem, on the RSA problem and on the selection of the redundancy function. The RSA problem[4], is finding an integer m such that $m^e \equiv d \pmod{n}$, where n is a product of two distinct large odd primes p and q, e is a positive random integer such that $\gcd(e,(p-1)(q-1)) = 1$ and an integer d. That is, the RSA problem is that of finding e^{th} roots of an integer d modulo a composite integer n.

The classical signature schemes, such as RSA, ElGamal and Rabin signature schemes, are described in the settings of the domain of integers Z. Many aspects of arithmetic over the domain of integers can be carried out to the domain of Gaussian integers Z[*i*], the set of all complex numbers of the form $a+bi$, where a and b are integers and to the domain of polynomials over finite fields $F[x]$[5]. Recently, the classical signature schemes were modified in many directions in these domains. El-Kassar *et al.*[6] modified the ElGamal signature scheme from its classical settings of the domain of natural integers to the domain of Gaussian integers by extending the arithmetic needed for the modifications to the domains. Similar extensions to the domain $F[x]$ was given by El-Kassar and Haraty in[7]. Haraty *et al.*[8] gave a comparative study of the extended ElGamal signature scheme algorithms. In[9], two extensions of the RSA signature scheme in the domain of Gaussian and the domain of polynomials over finite fields were presented. It was pointed out that the extended algorithms require a little additional computational effort than the classical one and accomplish much greater security.

In this study, we compare and evaluate the classical and modified RSA algorithms. We investigate the issues of complexity, efficiency and reliability by

Corresponding Author: Ramzi A. Haraty, Lebanese American University P.O. Box 13-5053 Chouran, Beirut, Lebanon 1102 2801

running the programs with different sets of data. Moreover, these different algorithms will be compared. In addition, implementation of an attack algorithm will be presented. Applying specific mathematical concepts to find the private key does this. After finding the key, it will be easy to sign the message. A study will be done using the results of running the attack algorithm to compare the security of the classical and modified signature scheme algorithms.

Classical and modified RSA signature schemes: The classical and modified RSA signature schemes are described in this section. Algorithms and examples are given. These algorithms are implemented to evaluate and compare the various methods.

Classical RSA signature scheme: In RSA signature scheme, entity A generates the public-key by first generating two large random odd primes p and q, each roughly of the same size and computing the modulus $n = pq$ and Euler phi-function $\varphi(n) = (p-1)(q-1)$[4]. Entity A then selects the exponent e to be any random integer in the interval (1, φ(n)) such that $\gcd(e,\varphi(n))=1$. Using the extended Euclidean algorithm for integers, entity A finds the exponent d which is the unique integer $(1,\varphi(n))$ relatively prime to $\varphi(n)$ such that $ed \equiv 1 \pmod{\varphi(n)}$. Hence, the public-key is the pair (n, e) and A's private-key is the triplet (p, q, d). A generates the signature as follows. First, entity A computes the redundancy function of the message m which is $m = R(m)$ such that $R(m)$ .Z_n and also computes $s \equiv m^d \pmod n$. Finally, A sends the signature s to entity B.
B validates the signature as follows. B obtains A's authentic public key (n, e), computes. $m \equiv s^e \pmod n$ and rejects the signature if m (M{R} (image of R). Finally, B recovers m by computing R(1(m).

Algorithm (RSA signature scheme):

* Find two large primes p and q and compute their product n = pq.
* Find an integer d that is relatively prime to φ(n) = (p(1)(q(1).
* Compute e from ed ≡ 1 (mod φ(n)).
* Broadcast the public key (n, e).
* Compute the redundancy function of the message m which is $m = R(m)$ such that $R(m)$ .Z_n.
* Sign the message m using the private-key by applying the rule $s \equiv m^d \pmod n$.
* The receiver validates the signature using the rule $m \equiv s^e \pmod n$.

Example: In order to generate the public-key, entity A selects the primes $p = 852225047$ and $q = 603309029$ and then computes the modulus $n = pq = 514155065595049363$ and the Euler phi-function $\varphi(n) = (p-1)(q-1) = 514155064139515288$. Next, A selects the exponent $e = 231814262079216429$ and uses the extended Euclidean algorithm for integers to find the exponent $d = 387883402970610381$ so that $ed \equiv 1 \pmod{\varphi(n)}$. Now, the public-key is the pair (n = 514155065595049363, e = 231814262079216429) and A's private-key is the triplet
(p = 852225047, q = 603309029, d = 387883402970610381).

To sign the message m = 1101100100111, for simplicity, take $R(m) = m$ so that R is the identity function. Then, $m = R(1101100100111) = 1101100100111$. A computes $s = m^d \pmod n = 1101100100111^{387883402970610381} \pmod{514155065595049363} = 502534570854711493$ and sends the signature 502534570854711493 to B. B obtains A's authentic public key (n = 514155065595049363, e = 231814262079216429), computes $m = s^e$ (mod n) $= 502534570854711493^{231814262079216429} \pmod{514155065595049363} = 1101100100111$ and computes $m = R^{-1}(m) = m = 1101100100111$.

RSA signature scheme in the domain of Gaussian integers: In RSA signature scheme, entity A generates the public-key by first generating two large random Gaussian primes β, γ and computes η = βγ. The Gaussian primes of Z[i] up to multiples of ±1 and ±I[10], are of the form: i) $\alpha = 1+i$; ii) $\pi = a+bi$ and $\overline{\pi} = a-bi$, where $q = \pi \cdot \overline{\pi} = a^2 + b^2$ is an odd prime integer of the form 4k+1; iii) p, where p is an odd prime integer of the form $4k+3$. If β and γ are selected to be of the form π and π, then the modified scheme is equivalent to the classical one[9]. If β and γ are selected to be of the form π and p, then η can be easily factored. Hence, β and γ are selected to be odd integers of the form 4k+3. Next, entity A computes $\varphi(\eta) = \varphi(\beta)\varphi(\gamma) = (\beta^2-1)(\gamma^2-1)$, where φ(η) is Euler phi-function in Z[i][11]. It selects a random integer e such that $1<e<\varphi(\eta)$ and e is relatively prime to φ(η). Then, entity A finds the unique integer d such that $ed \equiv 1 \pmod{\varphi(\eta)}$. A's public-key is (η, e) and A's private-key is (β, γ, d).

Represent the message as a number μ chosen from the complete residue system modulo η, $G_\eta = \{a+bi \mid 0 \le a \le \beta\gamma-1,\ 0 \le b \le \beta\gamma-1\}$. After computing the redundancy function of the message μ which is μ = R(μ), A computes the signature $s = \mu^d \pmod\eta$ and sends it to B. To verify the signature sent by A, B gets A's public key (η, e), computes the message representative μ as $\mu = s^e \pmod\eta$ and finally applies verification process to μ to recover μ. We note that the message space is enlarged so that its order is the square of that of the classical case. Also, more than the square of that of the classical case enlarges the range for the public exponent e. next; we provide three algorithms describing the RSA signature scheme over the domain of Gaussian integers. First, entity A generates the public and private keys by doing the following.

Algorithm (key generation for the RSA Gaussian signature):

* Generate two distinct large random Gaussian primes α and β, each roughly the same size.
* Compute $\eta = \alpha\beta$ and $\varphi(\eta) = (\alpha^2-1)(\beta^2-1)$.
* Selecte a random integer e, $1<e<\varphi(\eta)$ such that $\gcd(e, \varphi(\eta)) = 1$.
* Compute the multiplicative inverse d of e such that $ed \equiv 1(\bmod\ \varphi(\eta))$ using the extended Euclidean algorithm for Gaussian integers.
* Publish the pair (η, e) as the public key and keeping d as the private key.

Algorithm (Signature generation of RSA Gaussian signature):

* Represent the message as μ from the complete residue system modulo η, G_η.
* Compute $\mu = R(\mu)$ where $\mu.G_\eta$.
* Compute $s = \mu^d \pmod{\eta}$.
* Output s as the signature to B.

Algorithm (Signature verification of RSA Gaussian signature):

* Obtain A's authentic public key (η, e) .
* Recover $\mu \equiv s^e (\bmod\ \eta)$.
* Verify that $\mu.M_R$, otherwise reject the signature.
* Recover $\mu = R^{-1}(\mu)$.

Example: (RSA Gaussian signature scheme with small parameters): Public-Key Generation: Let $\beta = 91939$ and $\gamma = 69383$ be two Gaussian primes of the form $4k+3$. Compute the product $\eta = \beta\gamma = 6379003637$ and $\varphi(\eta) = (91939^2-1)(69383^2-1) = 40691687387592447360$. Entity A chooses $e = 25600002082007742863$ such that $\gcd(e, \varphi(\eta)) = 1$ and $1<e<\varphi(\eta)$. Using the extended Euclidean algorithm for integers, A finds $d = 33899823343652452847$ such that $ed \equiv 1(\bmod\ \varphi(\eta))$. Hence, A's public-key is the pair
$(\eta = 6379003637, e = 25600002082007742863)$ and A's private-key is the triplet $(\beta = 91939, \gamma = 69383, d = 33899823343652452847)$.

Signature Generation: To sign the message $\mu = 320177 + 147i$, for simplicity, take $R(\mu) = \mu$ so that R is the identity function and $\mu = R(320177 + 147i) = 320177 + 147i$. Afterwards, A computes $s = \mu^d = (320177 + 147i)^{33899823343652452847} \equiv 3059266386 + 5412724259i \pmod{6379003637}$ Finally, A sends the signature to B.

Signature verification: To validate the signature, B obtains first A's authentic public key $(\eta = 6379003637, e = 25600002082007742863)$. Then, B computes $\mu \equiv s^e \pmod{\eta} = (3059266386+5412724259i)^{25600002082007742863} \pmod{6379003637} = 320177 + 147i$ Finally, B computes $\mu = R^{-1}(\mu) = 320177 + 147i$.

The advantages of the RSA scheme in Z[i] are: first, generating two primes p and q in the form $4k+3$ in both the classical and the modified methods requires the same amount of effort. Second, the modified method provides more security than the classical method since the number of elements that can be chosen from to represent the message m is about the square of those used in the classical case. Therefore, we deduce that the extended RSA over the domain Z[i] provides an extension to the range of chosen messages, which make trials more complicated. The computations involved in the modified method do not require computational procedures that are different from those of the classical method.

RSA signature scheme over quotient rings of polynomials over finite fields: Let p be a prime number and let $h(x)$ and $g(x)$ be two distinct irreducible polynomials in $Z_p[x]$, the domain of polynomials over the finite field Z_p, where $h(x)$ is of degree s and $g(x)$ is of degree r. Let $f(x) = h(x)g(x)$. The polynomials $h(x)$ and $g(x)$ should be selected so that factoring $f(x) = h(x)g(x)$ is computationally infeasible. The quotient ring $Z_p[x]/<f(x)>$ is finite of order p^n, where $n = r+s$ is the degree of $f(x)$. It is well known that the quotient ring $Z_p[x]/<f(x)>$ is the direct sum of $Z_p[x]/<g(x)>$ and $Z_p[x]/<h(x)>$, that is $Z_p[x]/<f(x)> \cong (Z_p[x]/<g(x)>) \oplus (Z_p[x]/<h(x)>)$. Its group of units $U(Z_p[x]/<f(x)>)$ is the direct product of groups of units $U(Z_p[x]/<g(x)>)$ and $U(Z_p[x]/<h(x)>)$, that is $U(Z_p[x]/<f(x)>) \cong U(Z_p[x]/<g(x)>)\times U(Z_p[x]/<h(x)>)$.

Since $h(x)$ and $g(x)$ are irreducible, the quotient rings $Z_p[x]/<g(x)>$ and $Z_p[x]/<h(x)>$ are finite fields of order p^s and p^r, respectively. Hence, the groups $U(Z_p[x]/<g(x)>)$ and $U(Z_p[x]/<h(x)>)$ are cyclic with orders $\varphi(h(x)) = p^s-1$ and $\varphi(g(x)) = p^r-1$, respectively, so that $\varphi(f(x)) = (p^s-1)(p^r-1)$. We provide the algorithms of the extended RSA signature over polynomials. First, entity A generates the public and private keys by doing the following.

Algorithm (key generation for RSA signature over polynomials):

* Generate an odd prime p, two distinct monic irreducible polynomials $f(x)$ and $g(x)$ over Z_p.
* Compute $h(x) = f(x).g(x)$ in $Z_p[x]$.
* Compute the order of $U(Z_p[x]/<h(x)>)$ which is $\varphi(h(x)) = (p^s-1)(p^r-1)$.
* Select a random integer e where $1<e<\varphi(h(x))$ such that $\gcd(e, \varphi(h(x))) = 1$.
* Use the Euclidean algorithm for integers to find the unique multiplicative inverse d of e with respect to $\varphi(h(x))$ such that $1<d<\varphi(h(x))$ and $ed \equiv 1 \pmod{\varphi(h(x))}$
* Publish the key $(p, h(x), e)$ and keep d as private key.

To generate a signature on a message, entity A should do the following.

Algorithm (Signature generation of RSA signature over polynomials):

* Represent the message as a polynomial $m(x)$ in the complete residue system modulo $f(x)$ in $Z_p[x]$.

* Compute m(x) =R(m(x)), as a polynomial in the complete residue system modulo h(x) in $Z_p[x]$.

* Use the private key d to compute $s(x) = (m(x))^d$ (mod $h(x)$).

* Output $s(x)$ as signature of $m(x)$.

To verify the signature $s(x)$ and recover the real message $m(x)$, entity B should do the following.

Algorithm (Signature verification of RSA over polynomials)

* Obtain A's public key (p, $h(x)$, e).

* Compute $m(x) = s^e$ (mod $h(x)$).

* Verify that $m(x)$. M_R, otherwise reject the signature.

* Recover m(x) = $R^{-1}(m(x))$ where R^{-1} is the inverse of the Redundancy function.

Example: (RSA Signature Scheme over Polynomials with small parameters)

Public-Key Generation: Let p = 389. Entity A chooses the two irreducible polynomials $h(x) = x^2+376x+43$ and $g(x) = x^3+384x^2+3x+10$ in $Z_{389}[x]$. Reducing the polynomial $f(x) = h(x).g(x)$ in $Z_{389}[x]$ and computing $\varphi(f(x))$, A gets $f(x) = x,+371x.+111x^3+145x^2+388x+41$ and $\varphi(f(x)) = (389^3-1)(389^2-1) = 8907280505760$. Entity A then chooses the integer e = 95561135039 such that gcd(e, $\varphi(f(x))$) = 1 and $1<e<\varphi(f(x))$. Using the extended Euclidean algorithm for integers, A finds d = 5878808345759 satisfying $ed \equiv 1$ (mod$\varphi(f(x))$). Hence, A's public-key is (p=389, $f(x) = x,+371x.+111x^3+145x^2+388x+41$, e = 95561135039). and A's private-key is (d = 5878808345759, $g(x) = x^3+384x^2+3x+10$, $h(x) = x^2+376x+43$).

Signature generation: Choose $m(x) = 1+3x+x^2$ and assume that the redundancy function is the identity function (for simplicity). Thus, $m(x) =1+3x+x^2$. Afterwards, A computes
$s(x) = m(x)^d = (1+3x+x^2)^{5878808345759} \equiv 172x.+86x^3+265x^2+59x+177$ (mod $f(x)$) and sends $s(x)$ to B.

Signature verification: To validate the signature, B computes $m(x) = s(x)^e = (172x.+86x^3+265x^2+59x+177)^{95561135039} \equiv 1+3x+x^2$ (mod $f(x)$). So, $m(x) = 1+3x+x^2.M_R$. Hence, $m(x) = R^{-1}(1+3x+x^2) = 1+3x+x^2$.

RSA signature scheme attack: The security of the RSA signature scheme is based on the intractability of both the integer factorization problem and the RSA problem. Various attack schemes have been studied in the literature as well as appropriate measures to counteract these threats. Given the public-key, to forge the signature, a passive adversary must solve the RSA problem. There is no known efficient algorithm for this problem. One possible approach, which an adversary could employ, is to find the private key. In order to attack any protocol that uses the RSA signature scheme by finding its private key, the factorization problem must be solved first. After factorization, computing the value of Euler phi-function and then finding the private exponent d using the extended Euclidean algorithm for integers could solve the RSA problem. Once d is found, the signature can be forged. On the other hand, if the classical method is used and an adversary could somehow compute d, then n can efficiently be factored as follows[4]. This shows that in the classical case, the RSA problem and the integer factorization problem are computationally equivalent. It is not known if this remains true for the modified schemes. In the next section we evaluate the various RSA signature schemes by recovering the private key using the software package Mathematica. We illustrate the attack schemes in the following example.

Example: (Attacking the RSA signature scheme). Assume that the public key is: (n = 221806263006661919, e = 39786855994835377). To find the private key, we use the built-in Mathematica functions FactorInteger and PowerMod. The prime factors p and q are obtained from the output of FactorInteger[221806263006661919] which is {{315841909,1}, {702269891,1}}. Hence, p = 315841909 and q = 702269891. Next, we calculate $\varphi(n) = (p-1)(q-1) = (315841909-1)(702269891-1)=$ 404098131692231616. The exponent d = 279550294187496277 is the output of PowerMod[39786855994835377, −1, 404098131692231616]. The private key is (p = 315841909, q = 702269891, d = 279550294187496277).

Table 1: Running time in seconds: Classical RSA digital signature

Size of primes	Classical RSA Digital Signature		
	Public	Key Signature Verification	Public
50-digit	0.1341	0.002	0.006
100-digit	1.3801	0.011	0.0151
200-digit	4.2913	0.0471	0.0851
250-digit	5.7312	0.0923	0.1374
300-digit	7.3706	0.144	0.2074

Table 2: Running time in seconds: RSA Digital Signature with Gaussian integers

Size of primes	RSA Digital Signature with Gaussian integers		
	Public	Key Signature Verification	Public
50-digit	0.1341	0.0912	0.097
100-digit	1.3801	0.025	0.032
200-digit	4.2913	0.1101	0.1513
250-digit	5.7312	0.1883	0.2595
300-digit	7.3706	0.3035	0.4238

Table 3: Running time in seconds: RSA Digital Signature using polynomials

Prime p	Degree d	RSA Digital Signature using polynomials		
		Public	Key Signature Verification	Public
$p = 2$	$2 \leq d \leq 10$	0.0331	0.0161	0.0331
	$21 \leq d \leq 30$	1.7188	0.9222	1.6863
	$50 \leq d \leq 60$	15.6215	8.8147	17.211
$p = 101$	$2 \leq d \leq 10$	1.429	0.3365	0.4305
	$11 \leq d \leq 20$	8.992	3.1823	5.53
	$21 \leq d \leq 30$	45.1559	13.792	14.21275

Testing and evaluation: In this section, we compare and evaluate the different classical and modified signature schemes by showing the implementation of the signature schemes' algorithms with their running results. Also, we test the security of the algorithms by implementing different attack algorithms to crack the encrypted messages. All this is done using Mathematica 5.0 as a programming language and an Acer computer with Intel Pentium M715 processor, 1.5 GHZ CPU and 256 MB DDRAM.

RSA based digital signature algorithms: Using Mathematica 5.0 functions and an additional abstract algebra library, we have written programs for the following algorithms:

* Classical RSA digital signature.
* RSA digital signature with Gaussian integers.
* RSA digital signature with polynomials over a finite field.

After running the programs, it was clear that these programs have applied the RSA signature scheme in the correct way. All the programs have generated a public and private key with different mathematical concepts. Then a message is signed using the signature scheme and is sent to a verification procedure which returned the original message. The classical and Gaussian schemes were tested using the same public-key. The average running time of several runs using 50, 100, 200, 250 and 300-digit primes are given in Tables 1 and 2. The public-key was generated by randomly selecting odd integers having a given number of digits and of the form $4k$+3. The odd integers were tested for primality using the built-in Mathematica function PrimeQ until a prime is found.

To evaluate RSA algorithms using polynomials, we ran programs for various values of the prime p and degree of the irreducible polynomials. The average running time of several runs are listed in Table 3. The public-key was generated using the built-in Mathematica function IrreduciblePolynomial[x, p, d].

Comparing these algorithms, we conclude the following:

* All programs are reliable; they can sign, verify and return any message.
* The running time for the signature/verification algorithms is negligible in the classical and Gaussian cases. In the polynomial case the time for the signature/verification algorithms becomes significant for large primes and irreducible polynomials with large degree.
* The complexity for the three programs depends on the complexity of generating the public-key. Thus, the classical and Gaussian algorithms are equivalent since their public-key generation algorithms are identical when restricting the choice of primes to those of the form 4k+3. The Gaussian method is therefore recommended since the modified method provides an extension to the message space and the public exponent range.
* The public-key generation algorithm using polynomials requires the search for irreducible polynomials. The Mathematica built-in algorithm for generating irreducible polynomials appears to be inefficient as p becomes very large and the degree of the polynomial increases.

Attack algorithm: In order to attack any protocol that uses the RSA public key signature scheme by finding its private key, the factorization problem must be solved first. To test the security of the algorithms, we implemented attack schemes applied to the classical and modified signature scheme algorithms. For the classical and Gaussian algorithms, we generated public keys using primes of various sizes. The attack was conducted using the Mathematica built-in function FactorInteger to recover the prime factors. The Euler phi-function was then computed. Finally, the private exponent was obtained. The average running time of several runs are listed in Table 4.

Table 4: Attack time in seconds: Classical RSA digital signature

	Classical RSA Digital Signature				
Digits of p & q	20	22	24	26	30
Time	1.406	4.3983	26.3238	65.0656	94.245

For the RSA algorithms using polynomials, we generated a public-key using a prime p of various sizes and irreducible polynomials $f(x)$ and $g(x)$ of different degrees d. The attack was conducted by factoring $f(x)$ using the built-in function Factor[f, modulus->p] to recover the irreducible factors. The Euler phi-function was then computed. Finally, the private exponent was obtained. The average running time of several runs are listed in Table 5.

Table 5: Attack time in seconds: RSA Digital signature using polynomials

	Classical RSA Digital Signature			
Digits of p	5	5	10	22
Degree d	$10 \leq d \leq 11$	$12 \leq d \leq 13$	$5 \leq d \leq 6$	$2 \leq d \leq 3$
Time	2.373	2.954	0.651	0.231

After running these attack algorithms, we observed the following:

* All the attack programs are reliable so that they can sign any message by finding the private key.
* Attacking the classical and Gaussian RSA algorithms is easy if we are dealing with small prime numbers. However, when it comes to 100-digit prime numbers or higher, it needs about many computers working in parallel processing to compute the prime factorization of the multiplication of two 100-digit prime numbers.
* Attacking the RSA polynomial algorithm becomes more difficult as the size of p or the degree of the irreducible polynomials become larger.

CONCLUSION

In this work, we presented the classic RSA signature scheme and two of its modifications, namely, the RSA signature scheme in the domain of Gaussian integers, Z[i] and over quotient rings of polynomials over finite fields. We implemented these algorithms and tested their efficiency, reliability and security. The results obtained showed that all the algorithms applied the RSA signature scheme correctly and generated public and private key using different mathematical concepts. Messages were then signed using the signature scheme and were sent in encrypted form to a verification procedure which validated the signature and returned the original messages.

We also built attack scenarios directly aimed at solving the factorization problem. We modified the RSA attack algorithm to handle the modified algorithms. We observed that the Gaussian method is preferred since it is as secure as the classical one but provides an extension to the message space and to the signature exponent range.

As for future work, we plan to compare and evaluate the efficiency of the modified algorithms using very large numbers by using parallel computing techniques. We plan to run the programs in parallel on many computers and split the complex mathematical calculations between these computers. We plan to write a function that is capable of finding any random irreducible equation with respect to a specific prime number p. We also plan to apply the modified algorithms in many fields such as database, communications and network security.

REFERENCES

1. Diffie, W. and M.E. Hellman, 1978. New directions in cryptography. IEEE Trans. Information Theory, IT-22: 472-492.
2. Preneel, B., 1994. Cryptographic hash functions. Eur. Trans. Telecommunications, 5: 431-448.
3. Rivest, R., A. Shamir and L. Aldeman, 1978. A method for obtaining digital signatures and publickey cryptosystems. Communications of the ACM, 21: 120-126.
4. Menezes, A., J. Van Oorshot and P.C.S.A. Vanstone, 1997. Handbook of Applied Cryptography. CRC Press.
5. Smith, J.L. and J.A. Gallian, 1985. Factoring finite factor rings. Math. Mag., 55: 93-95.
6. El-Kassar, A.N., M. Rizk, N. Mirza and Y. Awad, 2001. ElGamal public-key cryptosystem in the domain of Gaussian integers. Intl. J. Appl. Math., 7: 405-412.
7. El-Kassar, A.N. and R. Haraty, 2005. ElGamal public-key cryptosystem in multiplicative groups of quotient rings of polynomials over finite fields. J. Computer Science and Information Systems, 2: 63-77.
8. Haraty, R., O. Otrok and A.N. El-Kassar, 2004. A comparative study of ElGamal based cryptographic algorithms. Proc. Sixth Intl. Conf. Enterprise Information Systems (ICEIS 2004), 3: 79-84.
9. El-Kassar, A.N., R. Haraty and Y. Awad, 2004. Modified RSA in the domains of gaussian integers and polynomials over finite fields. Proc. Intl. Conf. Computer Science, Software Engineering, Information Technology, e-Business and Applications (CSITeA'04). Cairo, Egypt.
10. Kenneth, A.R., 1988. Elementary Number Theory and its Applications. AT&T Bell Laboratories in Murray Hill, New Jersey.
11. Cross, J.T., 1983. The Euler's φ-function in the Gaussian Integers. American Mathematics Monthly, 90: 518-528.