International Journal of
Emerging Research in Management &Technology
ISSN: 2278-9359 (Volume-5, Issue-5)

Research Article | **May 2016**

Common Attacks on RSA and its Variants with Possible Countermeasures

Auqib Hamid Lone*, Prof. Moin Uddin
Department of Computer Science & Engineering,
Jamia Hamdard, New Delhi, India

Abstract—

Cryptography is used for secure communication since ancient days for providing confidentiality, integrity and availability to the information. Public key cryptography is a classification of cryptography having pair of keys for encryption and decryption. Public key cryptography provides security and authentication using several algorithms. RSA algorithm is prominent since its inception and is widely used. Several modified schemes were introduced to increase security in RSA algorithm involving additional complexity. In this paper we evaluate some common attacks on RSA and its variants and provide some necessary precautions to safeguard against such attacks.

Keywords— RSA, Encryption, Decryption, Attacks, Public key, Private Key

I. INTRODUCTION

In Today's digital world, communication is lifeline it has become as important as food, shelter and clothes for living day-to-day life. The challenge is to provide means for securing communication, so to allow people to communicate securely over insecure channel (medium). Information security plays important role in protecting digital information against security threats and keeps information secret by protecting it from unauthorized access. Information must be hidden from the unauthorized users(confidentiality), prevented from any modifications(integrity) and must be readily available to authorized users(availability). The value of information comprises confidentiality, integrity and availability,which are considered as three main goals of security and collectively called as CIA triad.There are several ways to implement these security goals, among which cryptography is most general and widely prevalent technique.

Cryptography and its Types

Cryptography is the study of mathematical tools and techniques related to aspects of informationsecurity such as confidentiality,data integrity, entity authentication, and data origin authentication. More precisely cryptography is the mathematical art of secret writing and is about the prevention and detection of cheating and other malicious activities.

Cryptographic Goals

1. *Confidentiality*is a service, which prevents the disclosure of private information to unauthorized users.
2. *Data integrity* is a service, which ensures data received by a receiver is the same as sent by sender. To assure data integrity, one must have the ability to detect data manipulation by unauthorized parties. Data manipulation may involve processes like insertion, deletion, and substitution.
3. *Authentication* is a service, which ensures the originality in terms origin, date of origin, data content, and time sent, etc. Authentication gives assurance that communicating entity is the one it claims to be
4. *Nonrepudiation* is a service that prevents either sender or receiver from denying a transmitted message. With Nonrepudiation receiver of message can prove that alleged sender in fact sent the message and vice versa.

Any cryptographic system mainly relies on three dimensions [1]:

- Operations involved in conversion of plain text to cipher text
- Number of secret keys used
- Method or algorithm of processing plain text

There are two types of cryptosystems based on the number of keys involved: Symmetric-key Cryptosystem (Secret key Cryptosystem) and Asymmetric-key cryptosystem (Public key cryptosystem).

Symmetric Key Cryptography

*In Symmetric key*Cryptosystem, encryption and decryption algorithm uses same key for conversion of plain text to cipher text and vice versa.

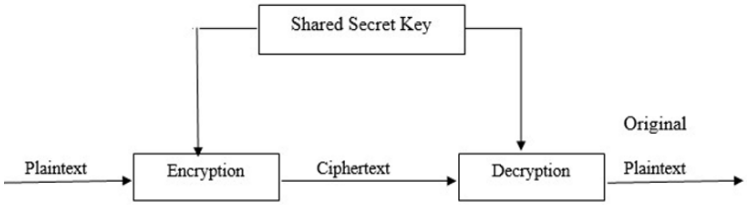

Figure 1: Symmetric key encryption/decryption scheme[2]

Asymmetric Key Cryptography

In Asymmetric key cryptography a pair of keys namely secret-key (or private key) and public-key is used for encryption and decryption process. Whitfield Diffie, Martin Hellman and Ralph Merkle introduced the concept of Asymmetric-key Cryptosystem [3]. Although different, the pair of keys are linked with mathematical function. The public key is used to encrypt plain text or to verify digital signature and private key is used to decrypt cipher text or to create digital signature.

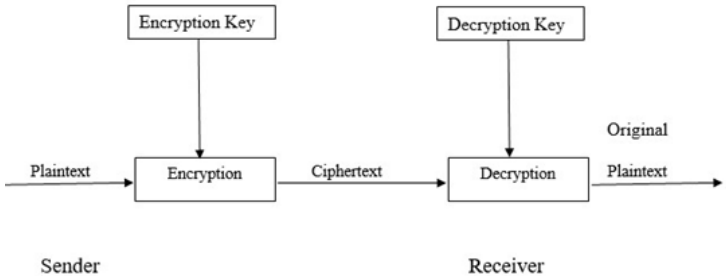

Figure 2: Asymmetric key encryption/decryption scheme[2]

Requirements for public-key cryptosystem

The requirements for public-key cryptosystem, which must be satisfied by public-key cryptographic scheme or algorithm, are listed below [1]:

- It is computationally easy for a party B to generate a pair of keys (public key, private key).
- It is computationally easy for a sender A to generate cipher text by applying encryption algorithm using public key on the message.
- It is computationally easy for the receiver B to retrieve original message by applying decryption algorithm using the private key on the cipher text.
- It is computationally infeasible for an opponent to determine the private key by only knowing the public key.
- It is computationally infeasible for an opponent to retrieve original message by only knowing the public key and cipher text.

Rest of the Paper is classified as follows section 2 gives brief overview of RSA Cryptosystem. In Section 3 we evaluate possible attacks on RSA and its variants with necessary precautionary measures and in section 4 we conclude our work

II. RSA CRYPTOSYSTEM

There were several schemes and algorithms proposed for Public key cryptography since its inception. One of the earlier known algorithm is RSA [4] named after its inventors R. Rivest, A. Shamir and L. Adleman. RSA algorithm was publicly disclosed in 1977 and widely used in secure data transmission. RSA found its use in securing communication over internet, providing confidentiality and authentication to e-mail and has become backbone for providing security to e-commerce. RSA is also fundamental part of various security protocols like SSL/TLS, SET, SSH, S/MIME, PGP, DNSSEC, as well as most PKI products [4]. It will not be wrong to say that RSA is present wherever security of digital data is prime concern.

The RSA Algorithm

The beauty of RSA algorithm lies in the fact that its underlying mathematical structure is quite simple and this is possibly the reason for its popularity. People generally feel more comfortable while playing with algorithm. It is build using basic algebraic operations on large integers. Before going into detailed description of algorithm first, we need to set some conventions: [5]

1. if *a*,*b* and *n* are positive integers, we say a is equal to *b* modulo *n* (denoted as $a \equiv b \bmod n$) if *b* is remainder of *a* divided by *n*;
2. We denote by $Z_n$the set of integers modulo *n*. This set can be represented by all non-negative integers less than *n*. We can define the operations addition and multiplication on this set by using usual operations on integers, but taking the result modulo *n*as defined above. These are called modular addition and modular multiplication.
3. We denote by *gcd(a ,b)*the greatest common divisor of *a*and *b*. This is defined by largest positive integer which divides both *a* and *b*.

We can now give detailed description of algorithm and is presented in the table:

Algorithm 2.1 RSA Algorithm

RSA_KeyGen( )
Input: Two large prime numbers p and q
Output:
Public key Components: {e, n}
Private key Components: {d, n}
Procedure:
n ←p*q

/* Compute Euler totient values of n */
$\Phi(n)$ =(p-1) *(q-1)
For random number e, satisfying 1<e<$\Phi(n)$ and *gcd* (e, Φ(n)) =1
Compute random number d, such that,
d ←e^{-1}mod Φ(n)
RSA_Encrypt()
Input: Plain text Message, M(<n)
Output: Cipher text, C
Procedure:

- B uses A's public key e to encrypt Message
  Cipher Text, C←M^emod n

RSA_Decrypt()
Input: Cipher text, C
Output: Decrypted plain text, M
Procedure:

- B transfers Cipher Text C to A
- A uses his private key d to recover original Message M

M←C^dmodn

Security of RSA

The security of RSA cryptosystem heavily relies on the very well-known problem of factoring large integers. The general idea behind the problem is that: it is fairly easy to find prime numbers and multiplying them together to get a single large number, However it is extremely difficult to start with a single large integer and find its prime factors and the problem is called as *Integer Factorization Problem.* Technically speaking this does not seem to be correct because it has never been mathematically proven that you need to factor *n*to recover *m* from *c* and *e* .One interesting point to note is that it has been shown that recovery of private exponent *d* is equivalent to factoring modulus *n* [6].That means if attacker gets (*e,d,n)* hecan efficiently factor *n* and can calculate private exponent *d.* In this case, we can say that he has completely broken the cryptosystem: he cannot only recover a particular message *M,* but also can decrypt all ciphertexts encrypted with the respective public key

Integer factorization problem is considered, as one of the hardest problems in mathematics because of the fact there is no polynomial time algorithm that solves this problem for maximum number of possible inputs. In fact factoring is not always hard, but a hard instance of the problem can be easily created, by simply multiplying two large chosen prime numbers. That is actually the trick that we use when working with RSA the other way of attacking the RSA cryptosystem is Brute force i.e. trying every possible *d* until attacker hits the correct one. Fortunately, the brute force attack is even less efficient than trying to factor *n.*

III. EVALUATION OF ATTACKS ON RSA AND ITS VARIANTS

Attacks on RSA are broadly classified into two categories: First Category includes those attacks that directly target the underlying mathematical function and Second category includes those attacks that exploit the flaws and weakness in its implementation. Let's us walk through with the attacks of both categories one by one in following subsections:

Attacks on RSA function

Attacks under this category mostly take the advantage of some special properties of RSA function. They usually exploit the misuse of the system, wrong choice for the selection of either private exponent *d* or public exponent *e*, relation between the ciphertexts, bad padding and many more. Most of these attacks use advanced mathematical techniques that are outside the scope of this thesis; praise worthy reference is [7].

- ***Low Decryption exponent attack against RSA***

RSA Decryption and signing computational complexity is directly proportional to the size of private exponent *d* and varies linearly with length of *d*. Most of the low-power devices tend to use small *d* in order improve computational efficiency. However attack by Michael Wiener shows that small *d* selection could result in complete breakdown of the cryptosystem. More specifically, he showed that if *n* is the modulus and *d* is the private exponent, with $d< 1/3(n)^{¼}$, then given the public key (*e, n*), an attacker can efficiently recover d (Boneh and Durfee have recently improved the bound to $d<n^{0.292}$). Speaking in practical terms, it is recommended for a typical 1024-bit RSA modulus, the private exponent should be at least 300 bits long. Interestingly if we choose *e*= 65,537 and calculate *d* from $ed \equiv 1 \bmod \Phi(n)$then it is guaranteed that we will get *d* of size comparable to *n* and consequently preventing this attack from posing threat to the cryptosystem.

Security tip:Choose large value of *d.*

- ***Partial Key Exposure Attack [5*]**

A well-known Cryptography principle says that the security of any cryptographic system should rely solely on the secrecy of the private key. An attack on the RSA cryptosystem proposed byBoneh, Durfee and Frankel shows the importance of protection of the entire private exponent d. They have shown that, if the modulus n is k bits long, given the (k/4) least significant bits of d, an attacker can reconstruct all of d in time linear to (e log(e)), where e is the public exponent. This means that if e is small, the exposure of a quarter of bits of d can lead to the recovery of the whole private key d.

Security tip: Use tamper resistant hardware modules or tokens for securely storing private key.

- ***Chosen Ciphertext Attack against RSA***

In this type of attack it is possible for an attacker to recover plaintext message *m* without the knowledge of private exponent *d* .To make things more clear lets us consider a scenario :

Eve is listening in on Alice's communications, manages to collect a ciphertext message *c*,encrypted with RSA with her public key. Eve's Aim is to recover plaintext message to be able to read the conversation. In terms of Mathematics Eve wants *m*, in which

$$m = c^d$$

To recover *m*, Eve first chooses a random number *r* , such that it is less than and relatively prime to *n*. She gets Alice's public key, *e*. Then she computes

$$x = r^e \bmod n$$
$$y = xc \bmod n$$
$$t = r^{-1 \bmod n}$$

if $x = r^e \bmod n$, then $r = x^d \bmod n$

Now Eve gets Alice to sign *y* with her private key, thereby decrypting *y*.(Alice has to sign the message not the hash of the message)Note Alice has never *y* before. Alice sends Eve

$$u = y^d \bmod n$$

Now everything is set, Eve computes

$$tu \bmod n = r^{-1}y^d \bmod n = r\text{-}^1x^dc^d \bmod n = c^d \bmod n = m$$

Finally, Eve got what she was looking for that is Plaintext message *m*.

Security tip: Never use RSA to sign a random document and always use one-way hash function first

- ***Common modulus attack against RSA***

Like in previous attack, an eavesdropper can under certain circumstances decrypt message sent without actually requiring a private key. For better understanding such an attack consider a scenario below:

An organization decides to use RSA for encryption and generates a public and private key pair for every employee. For making PKI (public key infrastructure) maintenance for the organization, it was decided that a single modulus n and each employee would be issued a personalized public key exponent *e*, and private key exponent *d*. Further organization selects encryption keys such *gcd* of any pair is 1

Suppose CEO dispatches the same message M for different office managers whose encryption exponents respectively are *e1* and *e2*. The encrypted messages are:

$$E = M^{e1}. \bmod n \text{ and } F = M^{e2} \bmod n$$

An eavesdropper has access to the public keys *n, e1,e2*and encrypted messages *E* and *F*. Since *gcd(e1,e2) = 1*, the eavesdropper applies the extended Euclidean algorithm to calculate integers *x and y* such *that x * e1 + y * e2 = 1* .Then *x and y* in hand the eavesdropper computes

$$E^x * F^y \bmod n$$

Interestingly this works out to be M and eavesdropper decrypted the message!

Security tip: One defense against this attack would be to ensure that the set of encryption exponents has number larger than 1 as a common factor.

- ***Blinding attack on RSA Signature***

Let us consider a scenario for better understanding of this attack.

Let d be Bob's private key and *(n,e)* be his corresponding public key. Suppose an adversary Eve wants Bob's signature on a message M, M <*n*. Being no fool, Bob refuses to sign M. Eve can try the following: she picks a random number r and computes M'= *re*·M mod *n*. She then asks Bob to sign the random message M'.Now Bob provides his signature on the innocent-looking M'. But recall that $S' = (M')^d \bmod n$.

Eve now computes S = S'/*r* mod *n* and obtains Bob's signature S on the original M. Indeed, $S^e = (S')^e/r^e = (M')^{ed}/r^e = M'/r^e = M \bmod n$. This technique enables Eve to obtain a valid signature on a message of her choice by asking Bob to sign a random"blinded" message. Bob has no information as to what message he is actually signing.

Security tip: Never use RSA to sign a random document and always use one-way hash function first

Implementation Attacks on RSA

All the attacks discussed in above section where acting on underlying cryptographic primitive and parameters. On the other side implementation, attacks are also called as side channel attacks target specific implementation details. In side channel attacks, attacker mostly uses some additional information leaked by the implementation of RSA function or exploits the faults in the implementation. These attacks mostly work against smart cards, security tokens. It is very hard to safe guard against such attacks, and only possible preventive measure is to reduce the amount of information leaked or make it irrelevant to attacker.

- ***Timing Attack***

P. Kocher introduced timing attacks against RSA in 1995 [8]. Timing attacks take advantage of the correlation between the private key and the runtime of the cryptographic operation. We know that the RSA private operation consists of a modular exponentiation, using the private key *d* as exponent. Modular exponentiations are usually implemented using an algorithm called repeated squaring algorithm. If the private key is *k* bits long, this consists of a loop running through the bits of *d*, with at most 2^k modular multiplications. In each step, the data is squared, with the execution of a modular multiplication if the current bit of the exponent is one. By measuring the runtime of the private operation on a

large number of random messages, an attacker can recover bits of d one at a time, beginning with the least significant bit. Note that in view of the partial key information attack described in earlier section, if a low public exponent is used, the attacker needs only to find the first k/4 bits using this method; the remaining bits can be found using the previous method.

Security Tip: To defend against timing attacks, one must try to lessen the correlation between the runtime and the private exponent. One solution is to add a delay, so that every modular operation takes the same fixed time. Another solution is called blinding. This transforms the data before the private operation using a random value generated by the cryptographic module

- ***Power Analysis***

P. Kocher, together with researchers from his company Cryptography Research, introduced in 1998 a new form of attack on smart cards and cryptographic tokens called power analysis. These attacks are mounted by monitoring the token's power consumption. Because the power consumption varies significantly during different steps of the cryptographic operation, an attacker can recover the secret information. They defined two types of attacks: *Simple Power Analysis* attacks work by directly observing a system's power consumption. *Differential Power Analysis* attacks are more powerful, using statistical analysis and error correction techniques to extract information correlated to private keys. Even though these attacks are quite complex and require a high level of technical skill to implement, Kocher says "they can be performed using a few thousand US dollars of standard equipment and can often break a device in a few hours or less" [9].

Security tip: These attacks are most effective against smart cards, and there are a number of techniques (both in hardware and in software) that can be used to prevent them.

- ***Fault Analysis Attack on RSA***

Analysis attacks work by exploiting errors on key-dependent cryptographic operations. These errors can be random, latent (e.g. due to bugs in the implementation) or induced. There are a number of fault analysis attacks against public-key and symmetric-key cryptographic devices. In 1997 Boneh, DeMillo and Lipton introduced an attack against RSA, which exploits possible errors on the RSA private operation in cryptographic devices [10]. As we know, the RSA private operation is a very compute-intensive operation, consisting of a modular exponentiation using numbers typically in the range of 300 decimal digits. Many implementations of RSA decryption and signing use a technique known as the Chinese Remainder Theorem (CRT), which by working modulo p and q (instead of module $n = pq$), can give a considerable improvement in the performance. Boneh, DeMillo and Lipton described a technique, which by exploiting an error occurring during the decryption or signing and analyzing the output, an attacker could factor the modulus n and therefore recover the device's private key. Both the output and the input of the operation are necessary for the attack to succeed (making it more effective against signing devices). To perform this kind of attack, one needs only to induce an error into the device during the private operation (for example, by voltage or clock speed variation). We should also note that unlike many of the attacks described before, the difficulty of this one is independent of the key length.

Security Tip: This fault attack against RSA can be easily prevented by requiring the device to verify the operation with the public key before outputting the result.

- ***Failure Analysis attack on RSA***

Failure analysis exploits feedback from the implementation indicating success or failure of the decryption operation. Attacks using failure analysis are generally adaptive chosen ciphertext attacks, and an application performing the decryption could be seen as an oracle that tests the validity of the transmitted ciphertext. An example of this type of attack was introduced by D. Bleichenbacher in 1998 and is known as the Million Message Attack [11].This attack exploits the cryptographic message syntax of some implementations.

Security tip: To prevent such an attack one should ensure to use an encoding system, which has strong security features.

Prevention and Countermeasures

Many tools and techniques have been used for cryptanalyzing RSA, resulting many clever and advance attacks. Although no such devastating attack has ever been found that breaks RSA cryptosystem completely. There are number of issues that require due care while working with RSA by users and developers. Points that require special attention are key size, properties of parameters (primes, exponents), and encoding and implementation details.

Key Size **-** As a rule, RSA keys should be chosen long enough, to thwart any attack and making system attack resistant. Several factors are taken into consideration while deciding the size of RSA key, important ones being: Value of data being protected, the expected lifetime of data, the threat model and best possible attack. Most of the current standards require the use of 1024 bits for RSA keys, since 512 RSA keys are proven too short and insecure for use.

Strong Primes **–** Apart from minimal length, some cryptographic standards also requires that primes used for RSA have some special properties. In particular, that *p-1* needs to have large prime factor. These so-called strong primes and plays heavy role in preventing Pollard's *p-1* factorization. But the fact is that, real security actually comes by choosing large enough primes *p* and *q*.

Multi-prime RSA **–** To speed up the RSA private operation, there are proposals of using more than two primes to generate the modulus. Using CRT, the speedup of an RSA system using *k*primes over the standard RSA is of around $k^2/4$. For example, a system deployed with 1024-bit modulus *n*, which is the product of 4 distinct (256-bit) primes, would perform around 4 times faster than the standard two-prime system. While the Number Field Sieve method cannot take advantage of this special form of *n*, 256-bit primes are currently considered within the bounds of the Elliptic Curve method (which is quite effective in finding small primes). Therefore, it is not recommended that 1024-bit moduli use more than three factors [12].

Public Exponent – In most RSA implementations, the public exponent e is usually chosen to be either 3 or 216 + 1 (= 65,537), with which the public operation takes 2 and 17 modular multiplications, respectively (instead of ~ 1000 multiplications expected for a randomly chosen *e*). Considering the existence attacks that we presented in earlier section, we believe 65,537 is more secure and should be used.

Private Exponent – From earlier section we believe, Wiener's low private exponent attack is quite effective, and if successful, can result in total recovery of the private exponent *d*. For the typical 1024-bit key, the private exponent should be at least 300 bits long. One should also pay special attention to the safeguarding of the private key. From partial key exposure attack, we came to know that the knowledge of a fraction of the key might allow the recovery of the entire key. The hardware solution is the most secure and should be considered whenever possible**.**

Encoding - The RSA algorithm applied to messages without any kind of preprocessing (known as raw RSA) offers a very weak level of security and should never be used. It is suggested that messages should always be encoded prior to encryption or signing. The study of properties of the encoding method to be used is an area of active research, as it has great impact in the overall security of the resulting cryptosystem. There are a number of different encoding methods defined, with allof them adding enough randomness as well as some redundancy. Special care is needed when using small public exponent and short padding. PKCS#1 [18] defines two encoding (padding) methods for encryption: PKCS#1v1.5 is a simple, ad-hoc design, which is widely deployed (SSL/TLS). Optimal Asymmetric Encryption Padding (OAEP) method is more complex, but has much better security properties.

Implementation – As discussed in section 4.2 most threatening and advance attacks against underlying mathematical structure of RSA arise due to carelessness in its implementation and one can probably say that currently the greatest threats to the security of the RSA cryptosystem are flawed implementations. In fact, the existence of side-channel attacks shows that extensive study of the mathematical structure of the RSA algorithm is not enough. Timing attacks are particularly effective, especially ones that exploit feedback from the implementation. Therefore, developers and designers of cryptographic applications and protocols need to pay special care and attention to implementation details, in particular leakage of information and error handling.

IV. CONCLUSION

The RSA cryptosystem is considered as the de facto standard for public-key encryption and signature worldwide.RSA is also fundamental part of various security protocols like SSL/TLS, SET, SSH, S/MIME, PGP, DNSSEC, as well as most PKI products [4]. It will not be wrong to say that RSA is present wherever security of digital data is prime concern. Till now no such devastating attack has ever been found and most problems that arise are due to be the result of misuse of the system, bad choice of parameters or flaws in implementations. In fact, years of research has built the trust of Security researchers on RSAand probably that are many reasons to believe that it will remain the most used public-key algorithm for years to come.

ACKNOWLEDGMENT

First of all, we are grateful to almighty Allah for giving us patience and perseverance for completing the work successfully.I take this opportunity to express my sincere thanks and deep gratitude to my friends who helped me in one or other way for completing my work successfully.

REFERENCES

[1] W. Stallings ,*Cryptography and Network Security: Principles and Practice*, 5th ed. Pearson Education, 2011.
[2] B. Schneier, Applied cryptography. New York [etc.]: Wiley-India, 2007.
[3] W. Diffie and M. Hellman, "New directions in cryptography", *IEEE Trans. Inform. Theory*, vol. 22, no. 6, pp. 644-654, 1976.
[4] R. Rivest, A. Shamir and L. Adleman, "A method for obtaining digital signatures and public-key cryptosystems", *Communications of the ACM*, vol. 21, no. 2, pp. 120-126, 1978.
[5] C. Cid, "Cryptanalysis of RSA: A Survey", *SANS Institute InfoSec Reading Room*, 2003. [Online]. Available: https://www.sans.org/reading-room/.../cryptanalysis-rsa-survey-1006. [Accessed: 02- Dec- 2015].
[6] B. Kaliski, *Timing Attacks on Cryptosystems*, no. 2, Redwood City, CA 94065 USA: RSA Laboratories, 1996.
[7] D. Boneh, "Twenty Years of Attacks on the RSA Cryptosystem",1999 *Notices of the American Mathematical Society(AMS)*,vol. 46,no. 2, pp. 203-213,1999.
[8] B. Kaliski, *Timing Attacks on Cryptosystems*, no. 2, Redwood City, CA 94065 USA: RSA Laboratories, 1996.
[9] "Differential Power Analysis - Rambus. Makers of Better.", Rambus. Makers of Better., 1998. [Online]. Available: http://www.cryptography.com/resources/whitepapers/DPA.html. [Accessed: 05- Jan- 2016].
[10] B. Kaliski and M. Robshaw, Comments on Some New Attacks on Cryptographic Devices, no. 5,Redwood City, CA 94065 USA :RSA Laboratories, 1997
[11] D. Bleichenbacher, B. Kaliski and J. Staddon, Recent Results on PKCS #1: RSA Encryption Standard, no. 7, Redwood City, CA 94065 USA:RSA Laboratories, 1998.
[12] D.Boneh and H. Shacham. Fast Variants of RSA. CryptoBytes. Volume 5, No. 1– RSA Laboratories.http://www.rsasecurity.com/rsalabs/cryptobytes/CryptoBytes_January_200 2_final.pdf

