International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 1 Issue 10, December- 2012

# Generate Dynamic Key On Asymmetric Key Cryptography Infrastructure

N. Yuvaraj, M.E[1], D. Manikandan, M.E,(Ph.D)[2], Dr. V. Parthasarathy[3]

[1]*Asst Professor, Dept of InformationTechnology, Excel Engineering College,Namakkal-TN*
[2] *Asst Professor, Dept of InformationTechnology, Excel Engineering College,Namakkal-TN*
[3] *Principal, Chettinad College of Engineering and Technology, Karur – Tamilnadu*

## ABSTRACT

**A new strong password key generation method is described. This algorithm provide fast and secure key generation in public cryptography world,(i.e. RSA algorithm)with help of random numbers (dynamic keys), without risk of offline and online dictionary attack. This new algorithm closely-related prime number generation techniques creating new pair of keys between authorized users. The class of secure password generating method provide safest and continues transaction between authorized users. These methods are important for several uses, ie reduce password guessing and knowing probability by others in other ways.**

Key Words: prime numbers, RSA, dynamic keys

## I. INTRODUCTION

The all computer systems are to which the logical access by a user (a human or another computer) is protected by a passwords. The threat that we consider is compromise of this password through one of the following two types of attack. In the on-line guessing type of attack, the attacker repeatedly makes guesses of the password, most likely first, and tests them by attempting to logon to the system. In our model the system has implemented "account lockout", locking the system after a certain number, say b, unsuccessful logon attempts which limits the effectiveness of the attack. In the off-line guessing type of attack, the attacker gets hold of some test-data from the system that enables him to test password guesses, most likely first, on his own systems. This information could for instance be a UNIX "passwd" file, a Windows SAM database or, more generally, the secure hash of a password. We distinguish two kinds of off-line attacks. In a complete attack the attacker is willing to take all the required computational effort to completely finish his attack algorithm thereby surely finding the password. In an incomplete attack the attacker is only willing to take a certain computational effort, a number of L guesses, in the attack, thereby finding the password only with a certain probability. To illustrate, suppose that an attacker has the SHA-1 hash of the password. If the attacker is willing to let the guess process run on a 1 GHz Pentium machine for a day this means that he is willing to perform about 236 tries ; one might find it acceptable that the probability of success is at most 1%.

The central problem of this paper deals with generating passwords that on the one hand have the functional requirement that they are "small" and on the other hand have the security requirement that they are "adequately" resistant against both on-line as off-line attacks, both complete as incomplete. Such passwords are specifically applicable in the context of one time passwords (e.g. initial passwords, activation codes).

## A. *Password Guessing and Cracking*

Attackers attempt to determine weak passwords and to recover passwords from password hashes through two types of techniques: guessing and cracking. *Guessing* involves repeatedly attempting to authenticate using default passwords, dictionary words, and other possible passwords. *Cracking* is the process of an attacker recovering cryptographic password hashes and using various analysis methods to attempt to identify a character string that will produce one of these hashes, thereby being the equivalent of the password to the targeted system. Guessing can be attempted by any attacker that can access the authentication interface, whereas cracking can only be attempted by an attacker who has already gained access to password hashes. This section describes guessing and cracking in detail and recommends strategies for mitigating these threat

## B. *Guessing*

There are several forms of guessing. In a *brute force attack*, the attacker attempts to guess the password using all possible combinations of characters from a given character set and for passwords up to a given length. This method is likely to take an extensive amount of time if there are many combinations to be tested. In a *dictionary attack*, the attacker attempts to guess the password using a list of possible passwords. The list may contain numbers, letters, and symbols, but is not an exhaustive list of all possible passwords or combinations that could create a password. In a *hybrid attack*, the attacker uses a dictionary that contains possible passwords and then uses variations through brute force methods of the original passwords in the dictionary to create new potential passwords. Since the attacker is adding characters and in some cases replacing characters based on a rule set in a controlled manner, the attack is more exhaustive than a dictionary attack but takes less time than a brute force attack. Another form of guessing attack is to search the victim's information for possible password content, such as family member names or birthdates.

## C. *Cracking*

Cracking involves attempting to discover a character string that will produce the same encrypted hash as the target password. The discovered string may be the actual password or another password that happens to produce the same hash. If the hash algorithm is weak, cracking may be much easier. Hash functions should be one-way, otherwise attackers that can access hashes may be able to identify passwords from them and successfully authenticate. Another example of a hash algorithm weakness is that some algorithms do not use salting. *Salting* is the inclusion of a random value in the password hashing process that greatly decreases the likelihood of identical passwords returning the same hash. If two users choose the same password, salting can make it highly unlikely that their hashes are the same

## D. *The concept of public key cryptography*

Two keys are used in public key cryptography. With help of two keys, we can reduces the probability of guessing or occur correct keys. With the public key one could encrypt messages, and Decrypt them with the private key. Thus the owner of the private key would be the only one who could decrypt the messages, but anyone knowing another idea that was observed was that of a key exchange. In a two-party communication it would be useful to generate a common secret key for bulk encryption using a secret key cryptosystem; public could send them in privacy. The public key cryptosystem algorithms have the following important characteristics

 It is computationally infeasible for a intruder to determine the decryption key given by the owner, even with knowledge of the cryptographic algorithm and the encryption key.

 Either one of the two related key can be used for encryption, while the other used for decryption.

## II. EXITING ALGORITHM

We have number of public key algorithm is available in this world for secure communication. The one of strongest algorithm is RSA public key cryptography. RSA has been widely used for establishing secure communication channels and for authenticating the identity of service providers over insecure communication mediums. In the authentication scheme, the server implements *public key authentication* with clients by signing a unique message from the client with its private key, thus creating what is called a *digital signature*. The signature is then returned to the client, which verifies it using the server's known public key

The algorithm flow is:

- To encrypt a message M the sender:
  - obtains **public key** of recipient KU={e,N}
  - computes: $C=M^e \bmod N$, where $0 \leq M < N$
- to decrypt the ciphertext C the owner:
  - uses their private key KR={d,p,q}
  - computes: $M=C^d \bmod N=(M^e)^d \bmod N=M^{ed} \bmod N$
  - note that the message M must be smaller than the modulus N (block if needed)
- From Euler's theorem
  - In RSA, n=pq and $0<m<n$, where p&q are prime numbers and n&m are integers
  - $m^{k\Phi(n)+1}=m^{k(p-1)(q-1)+1}$ where $\Phi(pq)=(p-1)(q-1)$
  - $Ed=k\ \Phi(n)+1$ ie $ed\equiv 1 \bmod \Phi(n)$ and $d \equiv e^{-1} \bmod \Phi(n)$

where, e with $\gcd(\Phi(n),e)=1$; $1<e<\Phi(n)$

## III. DYNAMICS KEYS

RSA is one of powerful algorithm with static prime numbers. RAS entire key establishment only based on two prime numbers. If once prime numbers identified or stolen by others, then all further message transaction will move to insecure channel. So in this new technique we create different set of keys ie prime numbers for secure all further transaction with help of new keys. Dynamic key establishment based on only prime number generation.

A dynamic key theory is described and analyzed. We discuss the security requirements for the sequence of dynamic keys and how they are used as a guide to build dynamic key generation functions. Based on that guide, we present a family of dynamic key generation functions. The dynamic key sequence created by this family of dynamic key generation functions is examined and analyzed. The analysis shows the advantages of dynamic keys in both security and efficiency. In the security analysis, we show that while one compromised dynamic key

exposes one message, the other messages in the session and system are still secure. Although perfect secrecy from one-time pad is impossible, the security of cryptographic system using dynamic key
is close to one-time pad. Besides minimizing cryptanalysis attack risks, dynamic keys are also able to prevent replay-attacks on authentication and payment systems.

In terms of performance, by storing intermediary keys, dynamic keys used as one-time symmetric cryptographic keys can achieve high levels of security without scarifying performance by increasing key size. Because the dynamic keys are generated online, there is no key exchange before every encryption. A study is conducted to find the most appropriate sequence size and dynamic key lifetime to balance between security and performance. Hence, the dynamic key generation scheme can adjust to suit different applications requiring different security levels.

## IV. GENERATING RANDOM PRIMES

### *A. Scope*

The goal is to generate a prime number q within the interval [$q_{min}$; $q_{max}$] in compliance with the ISO/IEC standard . In most cases, one has $q_{max} = 2^n - 1$, and $q_{min} = 2^{n-1} + 1$ when generating n-bit primes or qmin $= \sqrt{2^{2n-1}} + 1$ when generating 2n-bit RSA moduli of the form N = pq with p; q prime.

Our proposal consists in a pair of algorithms:the prime generation algorithm itself and an algorithm for generating invertible elements. We assume that a random number generator is at disposal along with a primarily testing oracle.

### *B. Prime generation*

Let $0 < \varepsilon \leq 1$ denote a quality parameter (a typical value for $\varepsilon$ is $10^{-3}$). The setup phase requires a product of primes $\Pi = \prod_i p_i$ such that there exist integers t,v,w satisfying

(P1) $1 - \varepsilon < \dfrac{\omega\pi - 1}{q_{max} - q_{min}} \leq 1$

(P2) $v\pi + t \geq q_{min}$

(p3) $(v+w)\pi + t - 1 \leq q_{max}$ and

(p4) the ratio $\Phi(\pi)/\pi$ is as small as possible.

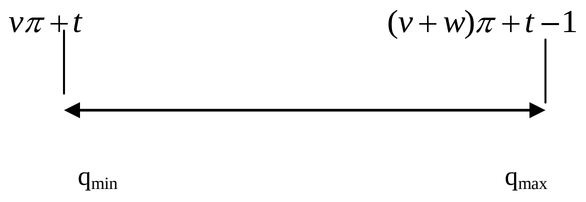

**Fig: Targeted Interval**

The primes output by our algorithm lie, in fact, in the sub-interval $v\pi + t$, $(v+w)\pi + t - 1 \subseteq$ [ $q_{min,}$ $q_{max}$] The error in the approximation is captured by the value of $\varepsilon$ : a smaller $\varepsilon$ gives better results. The minimality of the ratio $\Phi(\pi)/\pi$ in Property (P4) ensures that $\pi$ contains a maximum number of primes.

```
Input: parameters l = v$\pi$ ,m=w$\pi$ ,t and a $\Sigma$ $Z_m$

Output: a prime q $\Sigma$ [$q_{min}$,$q_{max}$]
```

1. Randomly choose k $\Sigma$ $Z_m$

2. Set q ← [(k - t) mod m] + t + l

3. If (q not prime) then

```
        (a) Set $k \leftarrow ak \pmod m$
        (b) Go to Step 2
4. Output q
```

**Fig : Prime Generation Algorithm**

It is worthwhile noticing that if a; k $\Sigma$ Zm so does their product, ak, since Zm is a (multiplicative) group. Therefore, throughout the algorithm, k remains coprime to m and thus to $\pi$ (remember that $\pi$ contains a large number of prime factors by (P4)). This, in turn, implies that q is coprime to $\pi$ as q $\equiv$ [(k -t) mod m]+t+l $\equiv$ k (mod $\pi$ and k $\Sigma$ $Z_\pi$ . Consequently, the probability that candidate q is prime at Step 3 is high.

We now specialize the previous algorithm to make it as fast as possible. The optimal value for t is t = 0.Moreover, it is advantageous to choose a so that multiplication by a modulo m is not very costly. The best possible value is a = 2. Unfortunately, 2 must belong to Zm owing to Condition (P4), 2 is a factor of $\pi$ and so of m, a contradiction. Our idea is to choose m odd (so that 2 $\Sigma$ $Z_m$) and to slightly modify the previous algorithm in order to ensure that prime candidate q is always odd. We require $\pi = \prod_i p_i$ (with pi $\neq$ 2) and integer's v and w (odd) satisfying:

(P2') $v\pi + 1 \geq q_{min}$

(P3') $(v + w) - 1 \geq q_{max}$

```
Input: parameters $l = v\pi$ and $m = w\pi$ (m odd)

Output: a prime $q \Sigma [q_{min}, q_{max}]$
```

```
1. Randomly choose $k \Sigma Z_m$
2. 2. Set q   k + l
3. 3. If (q even) then $q \leftarrow m - k + l$
4. If (q not prime) then
    (a) Set $k \leftarrow 2k \pmod m$
    (b) Go to Step 2
5. Output q
```

**Fig : Faster Prime Generation Algorithm**

Note that if k+l is even then m-k+l is odd since m- k+l $\equiv$ m $\equiv$ 1(mod 2). Hence, as before, a so-generated q is prime to 2 $\pi$ :gcd(q,2) = 1as q is odd; and gcd(q, $\pi$ )= 1 as q $\equiv$ ±k (mod $\pi$ and ± k $\Sigma$ $Z_\pi$

## V. RESULT AND DISCUSSION

This refined method provides more secured RSA public key cryptography with dynamic keys. This new method also provide accurate random numbers compare to previous normal implementation techniques, while occupying roughly the same amount of hardware resources. This new technique generates random numbers without the need of a secret seeds and also reduces the time requirement for generating random numbers. In application pseudorandom numbers produce subsets of these full sequences that are used as if they were drawn from some prime functions.

Some applications require that the primes found by our algorithm satisfy additional properties such as being strong or compliant with the ANSI X9.31 standard

This paper describes the various techniques available for the prevention of the denial of service attacks. They propose a new methodology along with the existing packet marking technique. The information contains the lifetime of the packet. Thus the packet marking scheme along with the embedding of the TTL value makes the traceback process an effective one.

## REFERENCES

[1] IEEE Std 1363-2000"IEEE Standard Specifications for public - Key Cryptography"IEEE Computer Society, August 29,2000.

[2] ISO/IEC WD18032"Prime number generation.*Working draft" April 18, 2001*

[3] Matsumoto,M.and Nishimura,T. Mersenne twister"a 623-dimensionally equidistributed uniform pseudo-random number generator" ACM Transactions on Modeling and Computer *Simulation (TOMACS) 8(1). 3-30.*

[4] Matsumoto, M. and Nishimura, T. Dynamic Creation of "Pseudorandom number generator. in Niederreiter, E.H. and Spanier, J. eds. *Monte Carlo and Quasi-Monte Carlo Methods"*, Springer, 2000, 56-69

[5] Shui Yu, Member, IEEE, Wanlei Zhou, Senior Member, IEEE, "Traceback of DDoS Attacks Using Entropy Variations", IEEE Transactions on Parallel and Distributed Systems, Vol. 22, No. 3, pp 412-425, March 2011.