Performance Estimates of cryptographic Hash vis-à-vis Secret Key and Public/Private Key operations

***Prof. Praveen Gupta**
**** Prof. Swapna** *K.*

Introduction

Cryptography is the lynchpin of data security- besides providing message confidentiality it facilitates message integrity, authentication and digital signatures. The design of a secure application is not possible without the use of some encryption and decryption techniques.
An important task for cryptographers is the analysis and recommendation of parameters, crucially including key size and thus implying key strength, for cryptographic primitives.
Cryptographic hash functions are important tool in cryptography to achieve certain security goals such as authenticity, digital signature, and digital time stamping and entity authentication. This paper presents an analysis of the speed and the performance of various security algorithms by varying key size and file size. Comparative study is based on symmetric encryption algorithm Data Encryption Standard (DES), asymmetric encryption technique RSA proposed by Rivest, Shamir, and Adelman and the widely used cryptographic hash function Secure Hash algorithm (SHA1).

Keywords: cryptography, hashing, RSA, DES, SHA1, performance estimates*.*

Encryption/Decryption

Symmetric-key cryptography is a mechanism by which the same key is used for both encrypting and decrypting. This characteristic requires sophisticated mechanisms to securely distribute the secret-key to both parties.DES is a 64 bit block symmetric cipher which means that it encrypts data 64 bits at a time.

Public-key on the other hand, introduces another concept involving key pairs: one for encrypting, the other for decrypting. Given a key pair, data encrypted with the public-key can only be decrypted with its private key; conversely, data encrypted with the private-key can only be decrypted with its public key. This characteristic is used to implement encryption and digital signature.

This concept has following advantages over symmetric-key:

• Simplified key distribution

• Digital Signature

• Long-term encryption

In 1977 Ron Rivest, Adi Shamir, and Leonard Adleman proposed a scheme using a public key for encrypting messages and a corresponding private key for decryption- this scheme is commonly referred as RSA- is based on a property of positive integers. It gets its security from the difficulty of factoring large numbers. However, it requires keys of at least 1024 bits for good security, which makes it quite slow. [Hook05]

Cryptography hash

A hash function is a deterministic function that maps an input element from a larger (possibly infinite) set to an output element in a much smaller set. A cryptographic hash function, h compress an input x of arbitrary length to a result with a fixed length i.e. h(x) =h'.

The properties of h are as follows:

1. One-way property: Given a hash value, y, it is computationally infeasible to find an input x such that h(x) =y.
2. Weak collision resistance: given an input value x1, it is computationally infeasible to find another input value x2 such that h(x1) =h(x2).
3. Strong collision resistance: it is computationally infeasible to find two input values x1 and x2 such that h(x1) =h(x2).

Confusion + diffusion: if a single bit in the input string is flipped, then each bit of the hash value is flipped with probability (approx.) 0.5.

Attack complexity

Weak collision resistance

Assume that the hash value is w bit long. So the total number of possible hash values is 2^w. Now a brute force attempt to obtain x would be to loop through the following operations

```
{
Do
{
Generate a random string, x'
Compute h (x')
}
While (h (x’))! =y)
Return (x’)
}
```

Strong Collision Resistance

```
{
// S is the set of (input string, hash value) pairs encountered so far
NotFound=true
While (notFound)
{
Generate a random string, x’
Search for a pair (x, y) in S where x=x’
If (no such pair exists in S)
        {
        Compute y’=h (x’)
        Search for a pair (x, y) in S where y=y’
        If (no such pair exists in S)
        Insert (x’, y’) into S
        Else
        NotFound=false
        }
}
return (x and x’)
}
```

This program terminates when the hash of a newly chosen random string collides with any of the previously computed hash values.

Generic Cryptographic Hash

All known hash functions are based on a compression function with fixed size input; they process every message block in a similar way. This has been called an “iterated” hash function. The information is divided into t b-bit blocks X1 through Xt. If the total number of bits is not multiple of the block length b, a padding procedure has to be specified. The hash function h with compression function or round function f can then be defined as follows:

$H0 = IV$

$Hi = f(Xi,Hi-1)$ $i = 1, 2, \dots t$

$h(X) = Ht$.

Here Hi are the intermediate variables or chaining variables that have a length of n bits, and the Xi are b-bit blocks. The result of the hash function is denoted with h(X) and IV is the abbreviation for Initial Value.

SHA-1

The compression function of SHA-1 uses the block cipher algorithm in feed forward mode to achieve one-wayness.

Performance Estimates

Private Key operations are generally slower than those with public keys. The ratio between private and public key operation times grows almost linearly with key length and longer keys provide an increased level of security at a performance cost [Manasce03]. Performance estimates of various algorithms are as follows:

1. The client and server programs for the **DES** security algorithm with a 64 bit key for different test file sizes was run.

   Test file sizes ranged from 1K to 10K, varying in size by 1K increments.The DES algorithm encryption and decryption times increase almost linearly as the file size increases. DES encryption times increased by a factor of 1.017 to 1.09 with each 1K increment in the test file size. The DES decryption times increased by a factor of 1.03 to 1.1 times with each 1K increment in the test file size. Like all symmetric key algorithms, the DES decryption times are 1.2 to 1.3 times faster than encryption times.

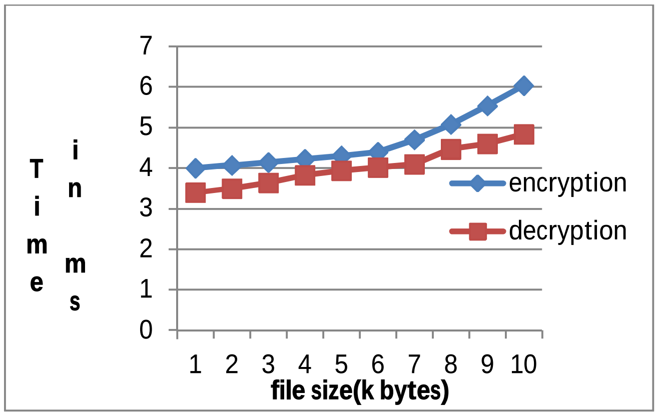

Fig 1.DES with 64 bit key

2. The client and server programs for the **RSA** security algorithm was run employing various test files and key sizes.

   Test file sizes included 100 bytes, 1K, 5K, and 10K.

   Key sizes included 1024 bits, 2048 bits, 3072 bits, 4096 bits, and 5120 bits.

   Encryption and decryption times on the client and server were measured and graphs were plotted representing measured times. According to the Figure 2 for every 1024 bits increment in key size, public key encryption times increased by a factor of 1.16 to 1.26 for the 100 byte file, by a factor of 1.32 to 2.72 for the 1K file, 1.53 to 2.73 for the 5K file, and 1.56 to 2.58 for the 10K file.

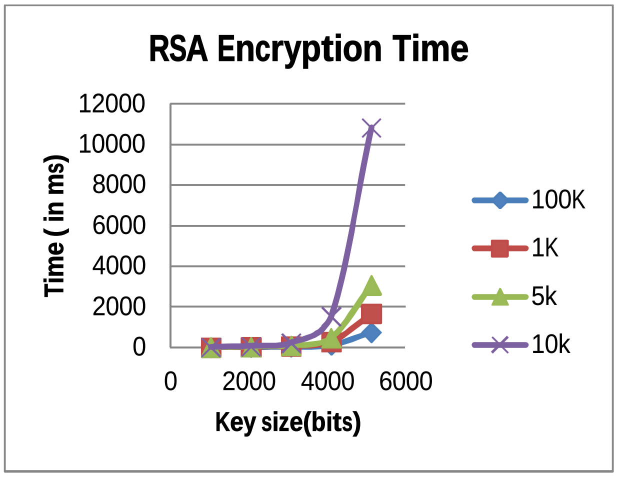

Fig 2 RSA encryption for varied file size

From fig 3 it can be concluded that for every increment of 1024 bits in key size, decryption times increased by a factor of 1.94 to 5.98 for the 100 byte file. The corresponding increase in decryption times is 1.9 to 6.4 for the 1K file, 1.9 to 6.8 for the 5K file, and 1.9 to 7.06 for the 10K file.

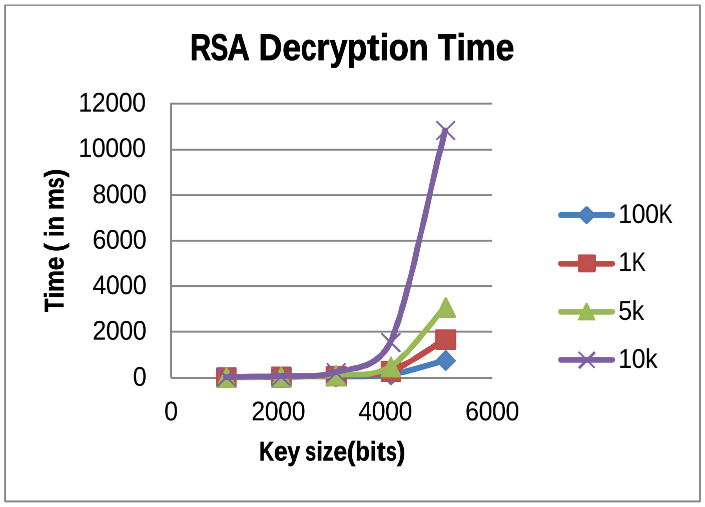

Fig 3. RSA decryption for varied file size

3. The client and server programs for digital signature with **SHA1** with various RSA key sizes were run. Key sizes included 512 bits, 768 bits, 1024 bits, and 2048 bits. A 1MB test file was used. Digital signature verification times on the client and server were measured and graphed. The digital signature verification times with SHA1 increased by a factor of 1.09 to 1.23 for each increment of RSA key size.

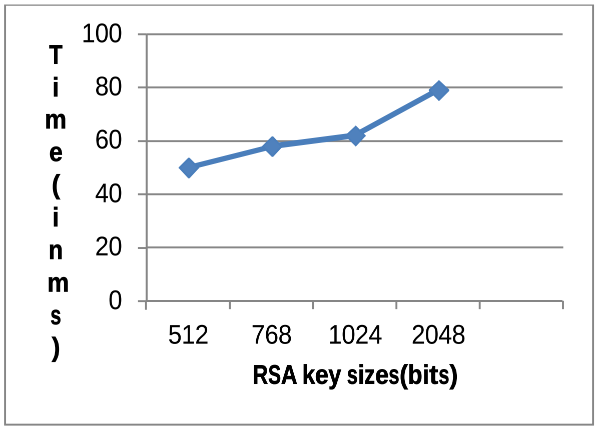

Fig 4. Digital signature time of SHA-1 with RSA

Table 1 compares the times to compute the cryptographic hash versus the times to perform encryption/decryption using 128 bit DES and 1024 bit RSA. The input file size in each case is 100 KB.

Table 1. Computation times for SHA-1, DES and RSA

SHA-1	DES (128 bit)		RSA (1024 bit)	
Cryptographic hash time(micro sec)	Encryption time (micro sec)	Decryption time (micro sec)	Encryption time (micro sec)	Decryption time (micro sec)
1844	3900	4000	520,000	10,020,000

It is clear that DES is more than twice as expensive compared to SHA-1. RSA is about three orders of magnitude more expensive compared to DES and SHA-1.

RSA decryption is much more expensive compared to RSA encryption.

Analysis of Test Results

When deciding upon key length in the design or use of a cryptographic algorithm, one must consider two tradeoffs. Long keys can provide more security but short keys can provide greater efficiency. Therefore, it is important to determine an optimal key length by evaluating the likelihood of the key being guessed, versus the impact of a longer key on the time required to encrypt the plaintext and the time required by the intended receiver to decrypt the ciphertext [Williams02].

References

- *[FIPS 180-1] Secure Hash standard. Federal Information Processing Standard publication 180, 1993.*
- *[BELL96] M. Bellare, R. Canetti and H. Krawczyk, "Keying hash functions for message authentication"/ Lecture Notes in Computer science, 1109, pp. 1-15, 1996. (Advances in Cryptology-CRYPTO'96)*
- *[GILB03] Henri Gilbert and Helena Handschuh, "Security Analysis of SHA-256 and Sisters", In: Mitsuru Matsui and Robert J. Zuccherato (Eds), "Selected Areas in Cryptography", LNCS, vol. 3006, pp.175-193, Springer, 2003.*
- *[WANGO5] Xiaoyun Wang, Hongbo Yu, "How to break MD5 and other hash functions", EUROCRYPT, 2005.*
- *[MERK89] R.C. Merkle, " A Certified digital Signature", Advances in Cryptology- CRYPTO'89 Proceedings, Lecture Notes in Computer Science, Vol. 435,G.Brassard(ed.), Springer-Verlag, 1989, pp. 218-238.*
- *[DAMG89] I. Damgard, " Adesign principle for hash functions", Advances in Cryptology-CRYPTO' 89 Proceedings Lecture Notes in Computer Science, Vol. 435, G. Brassard, Springer- Verlag, 1989, pp. 416-427.*
- *[YUVA79] G. Yuval, "How to swindle Rabin", Cryptologia (3), 1979, pp. 187-190.*
- *[RIVEE92] R. Rivest, "The MD5 message digest algorithm ", RFC 1321, MIT and RSA Data Security, Inc, April 19992.*