*IOSR Journal of Computer Engineering (IOSR-JCE)*
*e-ISSN: 2278-0661, p- ISSN: 2278-8727Volume 15, Issue 6 (Nov. - Dec. 2013), PP 27-33*
*www.iosrjournals.org*

# An Efficient implementation of PKI architecture based Digital Signature using RSA and various hash functions (MD5 and SHA variants)

Sankalp Prakash[1], Mridula Purohit[2]
[1]*(Computer Science & Engineering, Jagannath University, Jaipur, Rajasthan, India)*
[2]*(Mathematics Department, Vivekanand Institute of Technology (East), Jaipur, Rajasthan, India)*

***Abstract:*** *Digital Signature technique is widely being used to detect unauthorized modification to data and to authenticate the identity of the signatory. It is essential for secure transaction over unsecure/ open networks. Digital Signature schemes are mostly used in cryptographic protocols to provide services like entity authentication, authenticated key transport and key agreement. The PKI (Public Key Infrastructure) based digital signature architecture is related with RSA algorithm and secure Hash functions (MD5 &, SHA variants). RSA digital signature algorithm is an asymmetric cryptographic method whose security is associated with difficulty of factorization and hash function is applied to the message to yields a fixed-size message digest. This paper explores the PKI architecture based digital signature and presents an efficient way of its implementation and discusses various issues associated with signature schemes based upon RSA and hash functions. The results show that signing and verification are much faster in the developed application.*

***Keywords:*** *Digital Signature, MD5, RSA, SHA1, SHA2*

## I. Introduction

In this fast-paced technological world the importance of information and communication systems is escalating with the increasing significance and quantity of data that is transmitted to minimize operational cost and provide enhanced services. Unfortunately the vulnerability of systems and data are highly rising due to variety of threats, such as unauthorized access and use, destruction, alteration and misappropriation. Cryptography is the foundation of all data as well as information security aspects. A digital signature is an important type of authentication in the public key cryptographic system and it is widely used around the world. (Bruce Schneier, 1996; W.C.Cheng, C.F.Chou and L.Golubchik, 2002) .

By allowing the exchange of information more quickly, easily, and dependably than ever before, the Internet has forever changed the way of business and transactions. Electronic transactions are gaining in importance as nations around the globe because of significantly reducing the need for paper documentation while providing the opportunity for tremendous efficiency and productivity gains. As a result, digital signatures are poised to enter the mainstream as primary vehicle for establishing trust for a wide variety of electronic transactions. (Burton S. Kaliski, 2001; Rivest, Shamir, & Adleman, 1978) The information handled in electronic transactions is valuable and sensitive and must be protected against tampering by malicious third parties (who are neither the senders nor the recipients of the information). Sometimes, there is a need to prevent the information or items related to it (such as date/time it was created, sent and received) from being tampered with by the sender and/or the recipient. (S. R. Subramanya and Byung K. Yi, 2006)

A digital signature is a checksum which depends on the time period during which it was produced (Denning, 1984) . It is computed using a set of rules and a set of parameters such that the identity of the signatory and integrity of the data can be verified (Biometrics: the Future of Identification, 2000).

## II. PKI Architecture Based Digital Signature

The notion of digital signatures goes back to the beginning of public-key cryptography. In their landmark paper Whitfield Diffie and Martin Hellman (W.Diffie & M.E.Hellman, 1976) introduced the idea that someone could form a digital signature using public-key cryptography that anyone else could verify but which no one else could generate. After it RSA (Rivest, Shamir, & Adleman, 1978) has become the most proven and most popular, and achieved the widest adoption by standards bodies and in practice. (Burton S. Kaliski, 2001)

PKI is mainly used for secure transactions between companies or governmental agencies. An ecommerce Web site that uses SSL for encryption is a portion of PKI system. Encrypted e-mail is also another transaction that may be a part of a PKI system. Some companies or agencies may want all staff to digitally sign any documents they have created. Because a digital signature is derived from a Digital Certificate and its key, this is also part of a PKI system. There are so many possible scenarios and solutions it's almost impossible to list them all (Prakash Kuppuswamy, Peer Mohammad Appa and Saeed Q Y Al-Khalidi, 2012). PKI includes the mechanics described in this article as well as an ensemble of software, hardware and processes governed by rules and standards converging to the high level of Trust required and expected by the Industry. (CGI Group Inc., 2004) The RSA public-key cryptosystem and digital signature scheme are widely deployed today and have become essential building blocks for creating the emerging public-key infrastructure (PKI). (Burton S. Kaliski, 2001) In this paper the PKI (Public Key Infrastructure) based digital signature architecture has been discussed which is related with RSA algorithm and secure Hash functions (MD5 &, SHA variants).

Basically, the idea behind digital signatures is the same as handwritten signature which are traditionally used to validate and authenticate paper documents. A major difference between handwritten and digital signature is that a digital signature cannot be constant; it must be a function of the document that is sign. (Hemant Kumar, Ajit Singh, 2012) It is used to authenticate the fact that you promised something that you can't take back later. For electronic documents, a similar mechanism is necessary. Digital signatures, which are nothing but a string of ones and zeroes generated by using a digital signature algorithm, serve the purpose of validation and authentication of electronic documents. Validation refers to the process of certifying the contents of the document, while authentication refers to the process of certifying the sender of the document. (S. R. Subramanya and Byung K. Yi, 2006)

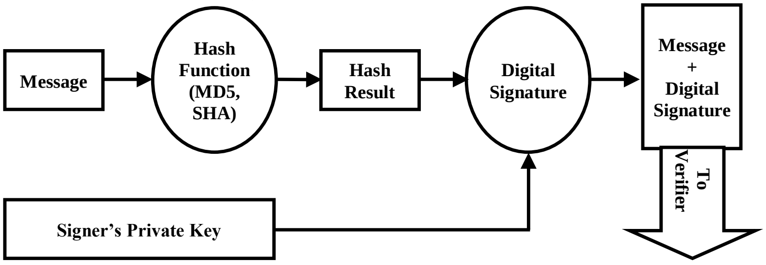
A digital signature is an electronic analogue of a written signature; the digital signature can be used to provide assurance that the claimed signatory signed the information. In addition, a digital signature may be used to detect whether or not the information was modified after it was signed i.e. to detect the integrity of the signed data. (Digital Signature Standard (DSS), June, 2009) In the situation where there is not complete trust between sender and receiver, something more than authentication is needed and the most attractive solution for this problem is the digital signature. Mainly digital signature is use in e-mail, electronic data interchange, software distribution, and other applications that require data integrity assurance and data origin authentication. The wireless protocols, like HiperLAN/2 (Martin Johnsson), and WAP (WAP Forum :), have specified security layers and the digital signature algorithm have been applied for the authentication purposes. (Hemant Kumar, Ajit Singh, 2012)

It must have some salient features such as verify the author and the date and time of signature; authenticate the contents at the time of signature; must be verifiable by third parties, to resolve disputes; it must be a bit pattern that depends on the message of being signed; must use some information unique to sender, to prevent both forgery and denial; must be relatively easy to produce to recognize and verify digital signature but computationally infeasible to forge it and must have legitimate concern.

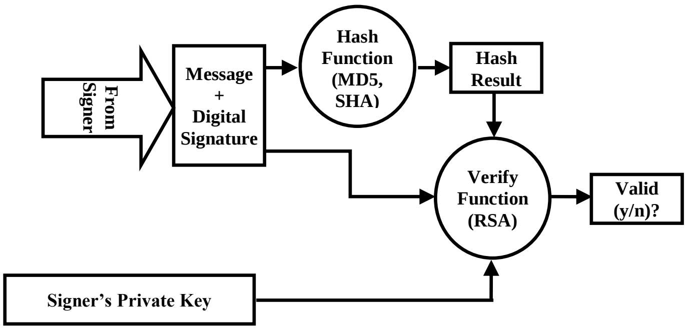
A digital signature can also be used to verify that information has not been altered after it was signed. A digital signature is an electronic signature to be used in all imaginable type of electronic transfer. Digital signature significantly differs from other electronic signatures in term of process and results. These differences make digital signature more serviceable for legal purposes.

Digital signatures are based on mathematical algorithms which includes a signature generation process and a signature verification process. A signatory uses the generation process to generate a digital signature on data; a verifier uses the verification process to verify the authenticity of the signature. These require the signature holder to have a key-pair (one private and one public key) for signing and verification. (Bruce Schneier, 1996; Rivest, Shamir, & Adleman, 1978; Digital Signature Standard (DSS), June, 2009)

Basic idea of digital signatures is each signer has a unique key called private key. There is also other part of key called public key. Whenever singer has to authenticate a document it creates a bit string called signature by applying his private key on the message or some hashed image of message as shown in Fig.1a. User who receives this message then applies his public key on the signature and checks the validity of the bit-string as shown in Fig.1b. If receiver is convinced that document is signed by legitimate signer, it accepts the document. Later if there is some dispute between sender and receiver regarding the validity of document, a third party inspects the signature and using the public key of signer verifies the signature. (Alfred J. Menezes, Paul C. van Oorschot and Scott A. Vanstone, 1997; William Stallings, Nov. 2005; Rania Salah El-Sayed, Moustafa Abd El-Aziem and Mohammad Ali Gomaa, 2008)



(a) Creating a Digital Signature



(b) Verifying a Digital Signature

**Figure 1 : Generalized signature Generation and Verification**

It has three phases namely (1) Key Generation (2) Signature Generation (3) Signature Verification. The Key Generation phase is the foundation phase for it.

1.1. Key-pair Generation

To generate key-pair for digital signature - RSA algorithm (Rivest, Shamir, & Adleman, 1978), most widely-used public key cryptography algorithm in the world, is used. The idea is that it is relatively easy to multiply prime numbers but much more difficult to factor. Multiplication can be computed in polynomial time where as factoring time can grow exponentially proportional to the size of the numbers. The algorithm is as follows:

a. Select p, q such that p and q both are primes and $p \neq q$.
b. Calculate n = p x q.
c. Calculate $\Phi(n) = (p - 1) \times (q - 1)$.
d. Select integer e such that $\gcd(\Phi(n),e)=1$ and where $1 < e < \Phi(n)$.
e. Calculate $d = e^{-1} \bmod \Phi(n)$. i.e. $ed = 1 \bmod \Phi(n)$
f. Public key KU = {e, n}.
g. Private key KR = {d, n}.

The Fig.2 shows the data flow diagram of key-pair generation in proposed RSAAPP.

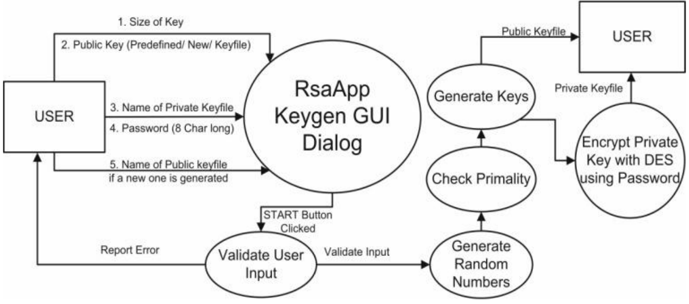


**Figure 2: DFD for Key-Pair Generation in RSAAPP**

**1.2. Signature Generation**

a. Given a message m, we apply a suitable hash function H (MD5, SHA1 or SHA2) to obtain the hash result M = H(m).
b. To sign a message *m*, we use M < n to compute Signature (S) = $M^d$ (mod n) where d is the private key of the signer.

The Fig.3 shows the data flow diagram of Signature Generation in proposed RSAAPP.

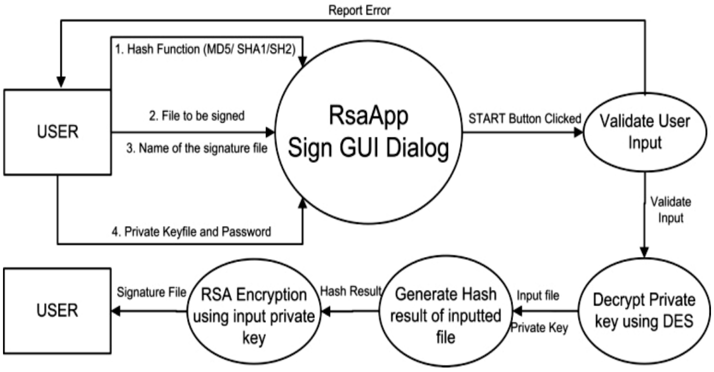

Report Error
1. Hash Function (MD5/ SHA1/SH2)
USER
2. File to be signed
3. Name of the signature file
RsaApp Sign GUI Dialog
4. Private Keyfile and Password
START Button Clicked
Validate User Input
Validate Input
Decrypt Private key using DES
Input file
Private Key
Generate Hash result of inputted file
Hash Result
RSA Encryption using input private key
Signature File
USER

**Figure 3: DFD for Signature Generation in RSAAPP**

### 1.3. Signature Verification

a. To verify the message *m*, we use the digital signature S to compute M = $S^e$ (mod n) where *e* is the public key of the signer.
b. Then we obtain M' = H(M) and compare it with M.
c. If both are same then the message is authentic otherwise it is tempered.

The Fig.4 shows the data flow diagram of Signature Generation in proposed RSAAPP.

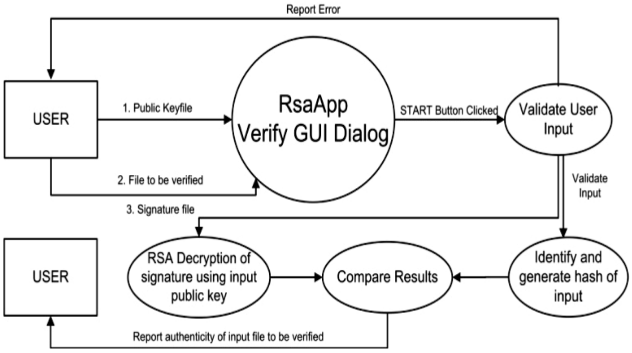


**Figure 4: DFD for signature Verification in RSAAPP**

## III. HASH FUNCTIONS (MD5, SHA1 AND SHA2)

A typical hash function takes a variable length message and produces a fixed length hash. Given the hash, it is impossible to find a message with that hash; in fact one cannot determine any usable information about a message with that hash, not even a single bit. Hash function are used to digest or condense a message down to a fixed size, which then be signed, in a way that makes finding other messages with the same hash extremely difficult (so the signature would not apply easily to other messages). Any cryptographic hash function H has 3 important properties: (1) given message P, it is easy to compute H(P) (2) given H(P), it is effectively impossible to compute P and (3) no one can generate two messages that have the same message digest.

### 1.4. MD5 Hash Function

The algorithm takes as input a message of arbitrary length and produces as output a 128-bit "fingerprint" or "message digest" of the input. It is conjectured that it is computationally infeasible to produce two messages having the same message digest, or to produce any message having a given prespecified target message digest. The MD5 algorithm is intended for digital signature applications, where a large file must be "compressed" in a secure manner before being encrypted with a private (secret) key under a public-key cryptosystem such as RSA. (Ronald L.Rivest) The summery of the MD5 hash function is:

F(x, y, z) = (x AND y) OR ((NOT x) AND z)
G(x, y, z) = (x AND z) OR (y AND (NOT z))
H(x, y, z) = x XOR y XOR z
I(x, y, z) = y XOR (x OR (NOT z))

### 1.5. SHA1 Hash Function

NIST, along with NSA, designed the Secure Hash algorithm (SHA1) for use with the digital signature standard. The algorithm published in 1995 in FIPS PUB 180-1 is commonly referred to as *SHA-1*.

When a message of any length < 2^64 bits is input, the SHA-1 produces a 160-bit output called a message digest. The message digest can then, for example, be input to a signature algorithm which generates or verifies the signature for the message. Signing the message digest rather than the message often improves the efficiency of the process because the message digest is usually much smaller in size than the message. The same hash algorithm must be used by the verifier of a digital signature as was used by the creator of the digital signature. Any change to the message in transit will, with very high probability, result in a different message digest, and the signature will fail to verify. (Donald E.Eastlake and Paul E.Jones, Sept. 2001) SHA-1 uses a sequence of logical functions, $f_0, f_1, \ldots, f_{79}$. Each function ft, where $0 \leq t < 79$, operates on three 32-bit words, x, y, and z, and produces a 32-bit word as output. The function $f_t(x, y, z)$ is defined as follows: (Secure Hash Standard, United States of America, National Institute of Science and Technology, Federal Information Processing Standard (FIPS) 180-1, April 1993)

$$f_t(x, y, z) = \begin{cases} \text{Ch}(x, y, z) = (x \text{ AND } y) \text{ OR } ((\text{NOT } x) \text{ AND } z) & (0 <= t <= 19) \\ \text{Parity}(x, y, z) = x \text{ XOR } y \text{ XOR } z & (20 <= t <= 39) \\ \text{Maj}(x, y, z) = (x \text{ AND } y) \text{ OR } (x \text{ AND } z) \text{ OR } (y \text{ AND } z) & (40 <= t <= 59) \\ \text{Parity}(x, y, z) = x \text{ XOR } y \text{ XOR } z & (60 <= t <= 79) \end{cases}$$

### 1.6. SHA2 Hash Function

In 2005, cryptanalysts found attacks on SHA-1 suggesting that the algorithm might not be secure enough for ongoing use. (Bruce Schneier, 2005) NIST required many applications in federal agencies to move to SHA-2 after 2010 because of the weakness.

SHA-2 is a set of cryptographic hash functions (SHA-224, SHA-256, SHA-384 and SHA-512) designed by the U.S. National Security Agency (NSA) and published in 2001 by the NIST as a U.S. Federal Information Processing Standard (FIPS) PUB 180-2. But in the proposed RSAAPP we use SHA-256.

SHA-256 uses six logical functions, where each function operates on 32-bit words, which are represented as x, y, and z. The result of each function is a new 32-bit word. (Secure Hash Standard, United States of America, National Institute of Science and Technology, Federal Information Processing Standard (FIPS) 180-2, 2002)

$$Ch(x,y,z) = (x \wedge y) \oplus (\neg x \wedge z)$$

$$Maj(x,y,z) = (x \wedge y) \oplus (x \wedge z) \oplus (y \wedge z)$$

$$\textstyle\sum_0^{\{256\}}(x) = ROTR^{2}(x) \oplus ROTR^{13}(x) \oplus ROTR^{22}(x)$$

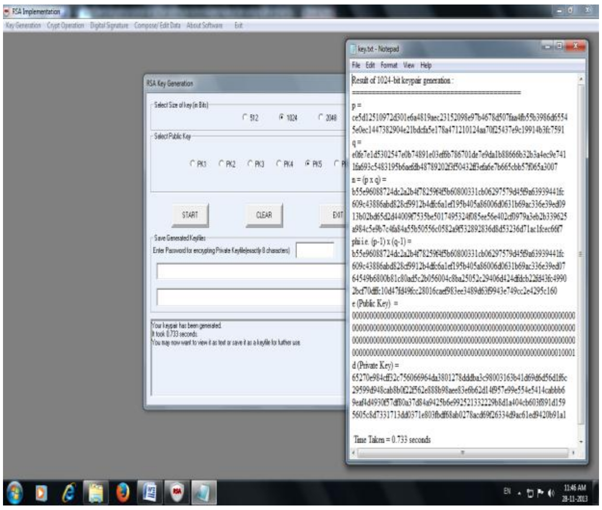
$$\textstyle\sum_1^{\{256\}}(x) = ROTR^{6}(x) \oplus ROTR^{11}(x) \oplus ROTR^{25}(x)$$

$$\sigma_0^{\{256\}}(x) = ROTR^{7}(x) \oplus ROTR^{18}(x) \oplus SHR^{3}(x)$$

$$\sigma_1^{\{256\}}(x) = ROTR^{17}(x) \oplus ROTR^{19}(x) \oplus SHR^{10}(x)$$

## IV. THE DEVELOPED APPLICATION (RSAAPP)

To test and compare the performance characteristics of the RSA and discussed signature algorithms, we developed application RSAAPP using C language with windows API (Charles Petzold, Nov. 1998). The key generation process generates the public and private keys in pairs. If required the keys can be viewed in hex format after generation. The corresponding private keys generated are encrypted using DES after taking an 8-character password as user input. Both keys are made read only. The proposed application can be used encrypt any kind of data i.e. text or binary. It uses SHA2, SHA1 and MD5 hash algorithms for the digital signature. SHA2 is much more secure then SHA1 and MD5. After key generation the Signature Generation module is processed for the file which is to be signed by the signer and sent to the recipient. Further at the receiving end the recipient verifies the signature by execution of Signature Verification process to authenticate that the file has been send by the authentic sender and to validate that the file has been tampered or not. The figures below shows the execution of the developed application i.e. RSAAPP-



Select Size of key (in Bits): ○ 512 ● 1024 ○ 2048

Select Public Key: ○ PK1 ○ PK2 ○ PK3 ○ PK4 ● PK5

Save Generated Keyfiles

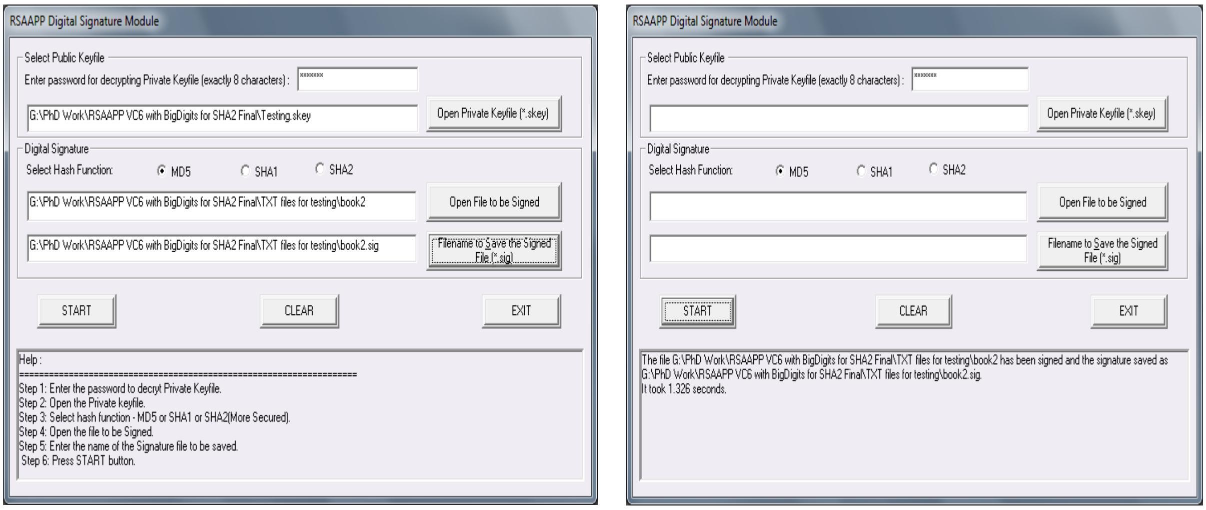
Enter Password for encrypting Private Keyfile(exactly 8 characters)

Your keypair has been generated.
It took 0.733 seconds.
You may now want to view it as text or save it as a keyfile for futher use.

```
Result of 1024-bit keypair generation :
=========================================================
p =
ce5d12510972d301e6a4819aec23152098e97b4678d507faa4fb55b3986d6554
5e0ec1447382904e21bdcfa5e178a471210124aa70f25437e9c19914b3fc7591
q =
e0fe7e1d5302547e0b74891e03ef6b786701de7e9da1b88666b32b3a4ec9e741
1fa693c5483195b6aefdb48789202f3f50432ff3efa6e7b665cbb57f065a3007
n = (p x q) =
b55e96088724dc2a2b4f78259f4f5b60800331cb06297579d45f9a63939441fc
609c43886abd828cf9912b4dfc6a1ef195b405a86006d0631b69ac336e39ed09
13b02bd65d2d44009f7535be5017495324f085ee56e402cf0979a3eb2b339625
a984c5e9b7c4fa84a55b50556c0582a9f532892836d8d53236d71ac1fcec66f7
phi i.e. (p-1) x (q-1) =
b55e96088724dc2a2b4f78259f4f5b60800331cb06297579d45f9a63939441fc
609c43886abd828cf9912b4dfc6a1ef195b405a86006d0631b69ac336e39ed07
64549b6800b81c80ad5c2b056004c8ba25052c29406d424dfdcb22fd43fc4990
2bcf70dffc10d47fd49fcc28016caef983ee3489d63f9943e749cc2e4295c160
e (Public Key)  =
0000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000010001
d (Private Key) =
65270e984cff32c756066964da3801278dddba3c98003163b41d69d6d56d1f6c
29599d948cab8b0f22f562e888b98aee83e6b62d14f957e99e554e5414cabbb6
9eaf4d4930f57df80a37d84a9425b6e992521332229b8d1a404cb603f891d159
5605c8d7331713dd0371e803fbdf68ab0278acd69f26334d9ac61ed9420b91a1

 Time Taken = 0.733 seconds
```

**Figure 5: Result of the Key Generation**


RSAAPP Digital Signature Module
Select Public Keyfile
Enter password for decrypting Private Keyfile (exactly 8 characters) :
Open Private Keyfile (*.skey)
Digital Signature
Select Hash Function:
● MD5
○ SHA1
○ SHA2
Open File to be Signed
Filename to Save the Signed File (*.sig)
START
CLEAR
EXIT
RSAAPP Digital Signature Module
Select Public Keyfile
Enter password for decrypting Private Keyfile (exactly 8 characters) :
Open Private Keyfile (*.skey)
Digital Signature
Select Hash Function:
● MD5
○ SHA1
○ SHA2
Open File to be Signed
Filename to Save the Signed File (*.sig)
START
CLEAR
EXIT

Enter password for decrypting Private Keyfile (exactly 8 characters) : *******

G:\PhD Work\RSAAPP VC6 with BigDigits for SHA2 Final\Testing.skey

Select Hash Function: ● MD5

G:\PhD Work\RSAAPP VC6 with BigDigits for SHA2 Final\TXT files for testing\book2

G:\PhD Work\RSAAPP VC6 with BigDigits for SHA2 Final\TXT files for testing\book2.sig

```
Help :
======================================================================
Step 1: Enter the password to decryt Private Keyfile.
Step 2: Open the Private keyfile.
Step 3: Select hash function - MD5 or SHA1 or SHA2(More Secured).
Step 4: Open the file to be Signed.
Step 5: Enter the name of the Signature file to be saved.
 Step 6: Press START button.
```

(a) Signature Generation module of RSAAPP

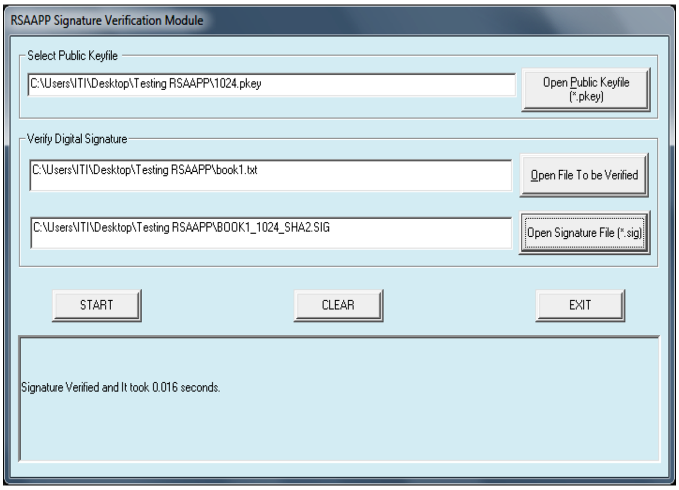
Enter password for decrypting Private Keyfile (exactly 8 characters) : *******

Select Hash Function: ● MD5

```
The file G:\PhD Work\RSAAPP VC6 with BigDigits for SHA2 Final\TXT files for testing\book2 has been signed and the signature saved as
G:\PhD Work\RSAAPP VC6 with BigDigits for SHA2 Final\TXT files for testing\book2.sig.
It took 1.326 seconds.
```

(b) Completion of Signature Generation of RSAAPP

**Figure 6: Signature Generation module of RSAAPP**


RSAAPP Signature Verification Module
Select Public Keyfile
Open Public Keyfile (*.pkey)
Verify Digital Signature
Open File To be Verified
Open Signature File (*.sig)
START
CLEAR
EXIT

C:\Users\ITI\Desktop\Testing RSAAPP\1024.pkey

C:\Users\ITI\Desktop\Testing RSAAPP\book1.txt

C:\Users\ITI\Desktop\Testing RSAAPP\BOOK1_1024_SHA2.SIG

```
Signature Verified and It took 0.016 seconds.
```

**Figure 7: Signature Verification module of RSAAPP**

## V. Experimental Results

The test results of the developed program RSAAPP for signature generation and signature verification using MD5, SHA1 and SHA2 are tabulated in seconds as shown in Table1. Three files of different size are chosen randomly from Calgary Corpus (Ian Witten, Timothy Bell and John Cleary, 2013) which is a collection of text and binary data files. Tests are preformed on an Intel P4 1.7GHz machine with 1GB of RAM with key size 1024 bits. Time taken in RSA 1024 bit key generation is 2.2 seconds. The experiment results with MD5, SHA1 and SHA2 hash functions are tabulated in seconds as shown in Table1.

**Table 1: Experimental Results**

| Filename with size | MD5 | | SHA1 | | SHA2 | |
|---|---|---|---|---|---|---|
| | **Sign. Gen.** | **Sign. Verify** | **Sign. Gen.** | **Sign. Verify** | **Sign. Gen.** | **Sign. Verify** |
| paper4 (12.9 KB) | 0.381 | 0.02 | 0.381 | 0.03 | 0.481 | 0.02 |
| news (368 KB) | 0.430 | 0.04 | 0.511 | 0.05 | 0.411 | 0.03 |
| book1 (596 KB) | 0.461 | 0.06 | 0.571 | 0.06 | 0.421 | 0.06 |

## VI. Conclusion

The experimental results shows that the RSA key (1024 bits), signature generation and signature verification with different hash functions – MD5, SHA1 and SHA2 are quite fast in developed RSAAPP. The cost of signature generation can be considered as a factor in the choice of signature system.

The developed RSAAPP system achieves high security for digital signature in addition to decrease processing time and computational overheads. And an intruder cannot pose the message sent since the sender's private key is unknown for him. Accordingly, the sender cannot be impersonated. On the receiver part, the message is verified by using sender's public key and his private key to decrypt the message successfully.

## Acknowledgements

The authors are extremely express gratitude to all those people and everyone who support directly or indirectly during the research work that "May god gives them long and prosperous life to spread the light of their intellectuality" and we are forever indebted for their efforts on my behalf, we have learned a great deal from them.

## REFERENCES

[1] Bruce Schneier, *Applied Cryptography Protocols, Algorithms, and Source Code in C*, 2nd ed. John Wiley & Sons, 1996.

[2] W.C.Cheng, C.F.Chou and L.Golubchik, "Performance of Batch-based Digital Signatures," in *10th IEEE International Symposium on Modeling* , 2002.

[3] Burton S. Kaliski, "RSA Digital Signatures," *Dr. Dobb's Journal*, May 2001, collabroation.cmc.ec.gc.ca/science/rpn/biblio/ddj/Website/articles/DDJ/2001/0105/0105c/0105c.htm.

[4] R. L. Rivest, A. Shamir, and L. M. Adleman, "A Method for Obtaining Digital Signatures and Public-Key Cryptosystems," *Comm. of the ACM*, vol. 21, no. 2, pp. 120-126, 1978.

[5] S. R. Subramanya and Byung K. Yi, "Digital Signatures," *IEEE Potentials*, vol. 06, pp. 5-8, Apr. 2006.

[6] D. E. Denning, "Digital signature with RSA and other Public-key cryptosystems," *Comm. of the ACM*, vol. 27, no. 4, pp. 388-392, Apr. 1984.

[7] "Biometrics: the Future of Identification," *IEEE Computer*, vol. 33, pp. 46-81, 2000.

[8] W.Diffie and M.E.Hellman, "New Directions in Cryptography," *IEEE Transactions Information Theory*, vol. 22, no. 6, pp. 644-654, Nov. 1976.

[9] Prakash Kuppuswamy, Peer Mohammad Appa and Saeed Q Y Al-Khalidi, "A New Efficient Digital Signature Scheme Algorithm based on Block cipher," *IOSR Journal of Computer Engineering*, vol. 7, no. 1, pp. 47-52, Nov. 2012.

[10] CGI Group Inc. (2004) Public Encryption and Digital Signature: How do they work?. [Online]. http://www.cgi.com/files/white-papers/cgi_whpr_35_pki_e.pdf

[11] Hemant Kumar, Ajit Singh, "An Efficient Implementation of Digital Signature Algorithm with SRNN Public Key Cryptography," *International Journal of Research Review in Engineering Science and Technology*, vol. 1, no. 1, pp. 54-57, Jun. 2012.

[12] "Digital Signature Standard (DSS)," National Institute of Standards and Technology FIBS PUB 186-3, June, 2009.

[13] Martin Johnsson. "HiperLAN/2 – The Broadband Radio Transmission Technology Operating in the 5 GHz Frequency Band", HiperLAN/2 Global Forum,Version 1.0 white-paper, 1999. [Online]. http://www.hiperlan2.com/technology.asp

[14] WAP Forum :. "Wireless Application Protocol Architecture Specification" and "WAP White Paper". www.wapforum.org.

[15] Alfred J. Menezes, Paul C. van Oorschot and Scott A. Vanstone, *Handbook of Applied Cryptography*. CRC Press, 1997.

[16] William Stallings, *Cryptography and Network Security Principles and Practices*, 4th ed. Prentice Hall, Nov. 2005.

[17] Rania Salah El-Sayed, Moustafa Abd El-Aziem and Mohammad Ali Gomaa, "An Efficient Signature System using Optimized RSA Algorithm," *International Journal of Computer Science and Network Security*, vol. 8, no. 12, 2008.

[18] Ronald L.Rivest. The MD5 Message-Digest Algorithm, RFC 1321, April 1992,. [Online]. http://www.faqs.org/rfcs/rfc1321.html

[19] Donald E.Eastlake and Paul E.Jones, "US Secure Hash Algorithm 1 (SHA1)," Sept. 2001, RFC 3174.

[20] Secure Hash Standard, United States of America, National Institute of Science and Technology, Federal Information Processing Standard (FIPS) 180-1, April 1993. [Online]. http://www.itl.nist.gov/fipspubs/fip180-1.htm

[21] Bruce Schneier. (2005, Feb.) Schneier on Security: Cryptanalysis of SHA-1. [Online]. https://www.schneier.com/blog/archives/2005/02/cryptanalysis_o.html

[22] Secure Hash Standard, United States of America, National Institute of Science and Technology, Federal Information Processing Standard (FIPS) 180-2, 2002. [Online]. http://csrc.nist.gov/publications/fips/fips180-2/fips180-2.pdf

[23] Charles Petzold, *Programming Windows*, 5th ed. Microsoft Press, Nov. 1998.

[24] Ian Witten, Timothy Bell and John Cleary. (2013) The Data Compression Resource on the Internet. [Online]. http://www.data-compression.info/Corpora/CalgaryCorpus/

**Sankalp Prakash** is a Research Scholar. He has completed M. Tech. in Computer Science from Rajasthan Vidyapeeth, Udaipur (Rajasthan), India in the year 2006 and pursuing PhD in Computer Science and Engineering from Jagannath University, Jaipur (Rajasthan), India. His major field of study is Cryptography and Computer Networks. Mr. Prakash is member of Computer Society of India (CSI), International Association of Engineers (IAENG) and Institution of Engineers (INDIA).

**Dr. Mridula Purohit** got post graduation degree in 1996 and doctorate in mathematics in 2000 from university of Rajasthan, Jaipur (Rajasthan), India. Her major field of study is Discrete Mathematic, Differential Equations, Special Functions, Polynomials, Cryptography and Communications. Presently she is a professor in Department of Mathematics at Vivekanand Institute of Technology (East), Jaipur (Rajasthan), India. She got more than 15 years of teaching experience. She has published more than 12 papers in International and National Journals and 04 text books. Recently she is working on research project titled "Applications of wavelet transforms in various fields of Science & Technology" awarded by All India Council for Technical Education, Government of India, New Delhi (India).