I.J. Wireless and Microwave Technologies, **2017, 5, 14-21**
Published Online September 2017 in MECS(http://www.mecs-press.net)
DOI: 10.5815/ijwmt.2017.05.02

***Available online at** http://www.mecs-press.net/ijwmt*

PRDSA: Effective Parallel Digital Signature Algorithm for GPUs

Dr. Sapna Saxena[a], Dr. Neha Kishore[b]

[a]*Chitkara University, Solan, Himachal Pradesh, India*
[b]*Chitkara University, Solan, Himachal Pradesh, India*

Received: 06 February 2017; Accepted: 18 June 2017; Published: 08 September 2017

Abstract

RSA based Digital Signature algorithm is an electronic scheme to ensure the security, authenticity and integrity of an electronic document intended to be used on Internet. Due to the involvement of RSA in signing and signature verification which is based on the series of modular multiplications and modular reductions on very large integers, the RSA based digital signature algorithm become compute-intensive and takes lot of time and energy to execute. A potential solution to this problem is to use the massive parallel powers of the multiprocessors of GPU that can simplify its complex computational part via CUDA programming. This paper presents a faster GPU based pRDSA algorithm which serves the same purpose as the RSA Digital Signature Algorithm but with less computational complexity. Proposed algorithm is an energy efficient parallel version of Digital Signature Algorithm to achieve high performance in the area of network security. It is based on SIMD model of parallel programming and implements repeated square-and-multiply method to compute the digital signature. The conceptual model of pRDSA and its performance have been discussed in this paper.

Index Terms: Digital Signature, RSA Algorithm, Graphical processing Unit, CUDA Programming, Parallel Computing

1. Introduction

Traditionally, the physical documents are authenticated by signatures which ensures its validity. It is not possible to tamper or forge the documents once they are signed. Moreover, the person who has signed the document cannot deny the signing at later stages. Similarly, the electronic documents can also be signed using digital signatures [1] to ensure their authenticity, integrity and non-repudiation.

* Corresponding author.
E-mail address:

RSA based Digital Signature [2] is an electronic scheme which is popularly used to provide security, authenticity and integrity to the electronic documents being shared during the vital online activities such as banking, transactions, e-commerce, etc. Practically, the digital signature algorithms are the combination of two sub-algorithms signing and verification. Signing and verification are based on two popular security algorithms SHA-1 for hash code generation and RSA for keys, signature generation and verification. Due to the involvement of RSA [3] which is based on the series of modular multiplications and modular reductions [4] on very large integers, the digital signature algorithm become compute-intensive and takes lot of time and energy to execute.

The most powerful solution to this problem is to execute the complex computational parts of the algorithm in parallel. To accomplish this the Digital signature algorithm is redesigned using CUDA programming model to use the massive parallel [5, 6] powers of the multiprocessors of GPUs. In this paper, the redesigned parallel Digital Signature algorithm pRDSA is proposed for secure implementation of digital signature on GPUs to reduce power and time consumption and achieve high performance. The redesigned algorithm is implemented on GPU enabled machine and the promising results are achieved. The framework of the parallel algorithm and its results are also discussed in the paper.

2. Background

This section provides the fundamental information of the technologies being discussed in this paper.

2.1. Digital Signature Algorithm

A Digital signature [7] is mathematical scheme which is used to ensure the security, authenticity and integrity of an electronic document meant for transmitting on the network. The main characteristic of the digital signature is non-repudiation according to which if the document is digitally signed, the sender cannot deny that he has sent the message to the recipient. In other words, a digital signature gives a recipient an authentic reason to consider that the message was created by which sender [8]. Digital signatures are used in online money transactions, software distribution, and to detect tampering forgery. Digital Signature ensures following security features –

- **Authentication:** The document is the one that was originally sent by the sender.
- **Integrity:** The document is the original one and nobody has tampered with it during the transmission.
- **Non-Repudiation:** The sender cannot deny at later stages that he or she has not send the message.

2.2. CUDA Programming for GPUs

In November 2006, Nvidia Corporation has introduced a general purpose parallel computing architecture to utilize the massive parallel computing capabilities of GPUs. Compute Unified Device Architecture (CUDA) [9] is a parallel programming model and its instruction sets are used to leverage the parallel powers of NVIDIA GPUs to solve complex mathematical computations based scientific problems in a more efficient way. CUDA allows its developers to use C as a base programming language to develop powerful parallel applications. In addition to C, other languages such as FORTRAN, C++ also supports CUDA programming [10] environment.

3. Related Work

This section provides some references to previous work related to developing parallel digital signatures solutions on the GPUs.

In 2008, Robert Szerwinski and Tim G¨uneysu presented a paper “Exploiting the Power of GPUs for Asymmetric Cryptography” [11]. In their paper, the authors focused on the efficient realization of the

computationally expensive operations in asymmetric cryptosystems. They proposed novel implementations employing GPUs as accelerator for RSA and DSA cryptosystems as well as for Elliptic Curve Cryptography (ECC). In their implementations, they were able to compute 813 modular exponentiations per second for RSA or DSA-based systems with 1024 bit integers.

In the paper titled "NTRU Modular Lattice Signature Scheme on CUDA GPUs" [12], authors Wei Dai et al proposed how to use Graphics Processing Units (GPUs) with Compute Unified Device Architecture (CUDA) to accelerate a lattice based signature scheme, namely, the NTRU modular lattice signature (NTRU-MLS) scheme. They obtain a 2× improvement in the signing speed; for the revised parameter sets, where acceptance rate of rejection sampling is down to around 1%, our implementation can be as much as 47× faster than a CPU implementation.

In 2010, Lin, Zhao and Yang presented a paper "A CUDA Based Implementation of an Image Authentication Algorithm" [13] where signatures and watermarking were used for image authentication. In this paper, they proposed a CUDA-based implementation of image authentication algorithm with NVIDIA's Tesla C1060 which is found to be 19-36 times faster than the original implementation in CPU.

4. PRDSA: Parallel Digital Signature Algorithm

The pRDSA is a time and energy efficient parallel algorithm that serves the same purpose as the digital signature algorithm. It is the combination of Signing and Signature verification sub-algorithms where SHA-1 is used to generate the MAC (Message Authentication Code) for the message and the RSA algorithm is used to generate private and public key. It incorporates memory-efficiency method, repeated square-and-multiply modular exponentiation algorithm [14, 15] which is based on following mathematical formula:

$$base^{exp} \; mod \; modulus = \; base^{exp/2} * \; base^{exp/2} \; mod \; modulus$$

Algorithm 1. pRDSA Signing Algorithm

```
Procedure: signature_generation
Model: Thread Model based on repeated square-and-multiply
Output: Signature
Read Message from File to a Vector and Apply SHA-1 to Generate HashCode
Apply private key on Hashcode to Generate Signature as:
    Declare N := Power / Number_Of_Blocks
    Divide the whole procedure into N parts
    Declare No_Of_Threads for  N no. of Par_Sec
    Parbegin for each thread
        Declare Result:=1
        If Power mod 2 := 0
         for j := 1 to N
            Assign Result := Result * Base
         end for
      else
         for j := 1 to N
            Assign Result := Result * Base
         end for
      Assign Result := Result * Base;
      end if
Parend
Assign Signature := Result mod Modulus
Concatenate Signature to Message
```

PRDSA is data parallel in nature and based on SIMD architecture. To obtain maximum degree of concurrency multiple modular multiplication modules are mapped with different processing units of GPU concurrently. The data decomposition technique is used to divide and map the tasks among parallel cores in

order to make optimum use of resources is applied to the algorithm. The structure of pRDSA is illustrated in Fig 1.

The pRDSA is a combination of two parallel sub-algorithms: parallel digital signing algorithm and parallel signature verification algorithm. The parallel digital signing algorithm is based on SIMD (Single Instruction Multiple Data) model given as Algorithm 1. For digital signing, first the SHA-1 hashing algorithm is applied to the message to generate its hash code. Thereafter, the hash code generated is signed with the RSA private key in parallel by doing data decomposition to generate the signature. The signature is concatenated with the message and transmitted along with public key to the receiver. In certain cases, the public key is shared explicitly with the receiver.

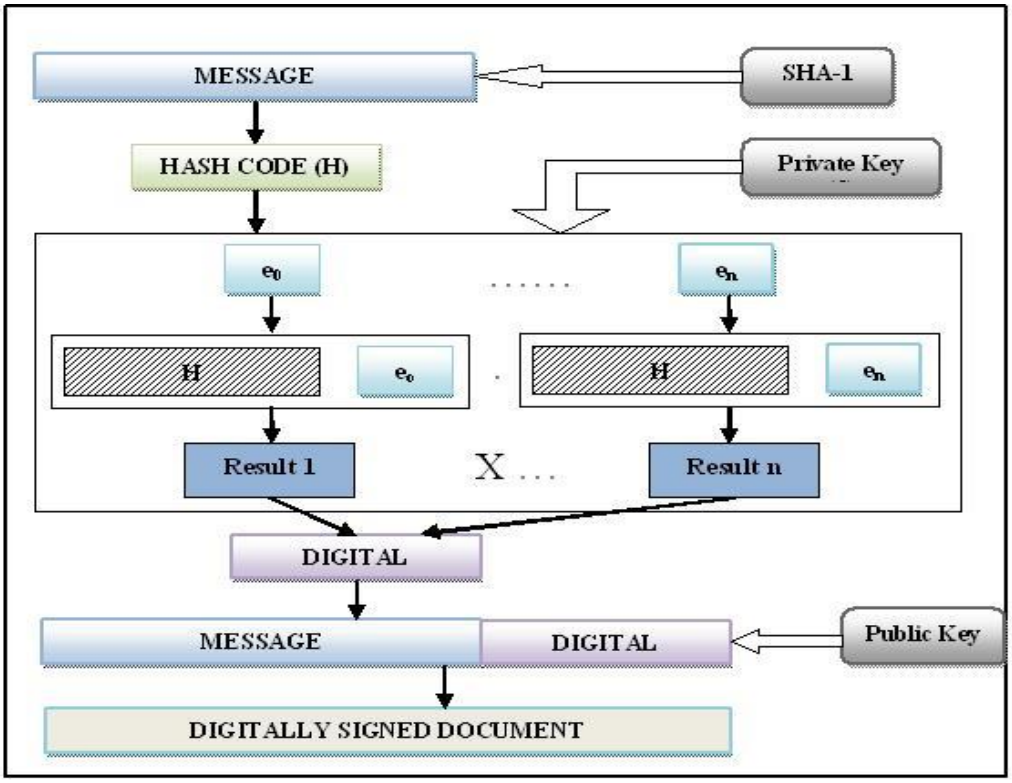

Fig.1. pRDSA Structure

The parallel signature verification algorithm is used at the receiver end. It is based on SIMD (Single Instruction Multiple Data) model given as Algorithm2. For signature verification, first the signature is extracted from the message and thereafter RSA public key is applied using parallel signature verification algorithm to it to get the original hash code. Then the SHA-1 is used to generate the received message's hash code. Now both the hash codes are matched in order to prove the authenticity of the message. If codes are found to be same then the message is considered to be authentic and used. In the pRDSA algorithm the whole process of signature generation and verification is done in parallel.

Algorithm2. pRDSA Signature Verification Algorithm

```
Procedure: signature_verification
Model: Thread Model based on repeated square-and-multiply
Output: Signature Verification
Extract Signature from the message and Apply SHA-1 to Generate HashCode
Apply public key on Signature to get original hashcode:
Declare N := Power / Number_Of_Blocks
    Divide the whole procedure into N parts
    Declare No_Of_Threads for N no. of Parallel Section
    Parbegin for each thread
        Declare Result:=1
        If Power mod 2 := 0
         for j := 1 to N
            Assign Result := Result * Base
         end for
      else
         for j := 1 to N
            Assign Result := Result * Base
         end for
      Assign Result := Result * Base;
      end if
Parend
Assign Signature := Result mod Modulus
Verify Both Hashcodes                end for
      Assign Result := Result * Base;
      end if
Parend
Assign Signature := Result mod Modulus
Concatenate Signature to Message
```

5. Platform Used

In order to test the performance of pRDSA it is implemented using CUDA programming on GPU enabled machine. The configuration of the target device is given below:

Processor: AMD FX (tm) - 8120 Eight- Core Processor
RAM: 8.0 GB
O.S.: Ubuntu Linux 11.04
Environment: gcc 4.5.4
GPU: Tesla C2075
API: CUDA 5.5

6. Result Analysis and Discussions

The algorithm is tested using series of experiments involving the RSA keys having varied key sizes ranging from 512 bits to 2048 bits. The thread size and the block size have been taken as per the key size. The results observed during experiments are presented in the Table 1 and Fig 2. The promising speedup of 3x to 5x has been recorded for various test cases while executing the algorithm on GPUs. It can also be observed in the results that as and when the key size is increased the better speedup has been obtained, which assure the strength of the redesigned algorithm for larger key sizes.

Table 1. Execution Time (In Seconds)

Key Size	Sequential RSA-DSA on CPU	pRDSA on GPU
RSA512	0.0262	0.00844
RSA1024	0.1551	0.041
RSA1280	0.27532	0.065
RSA1536	0.54916	0.11414
RSA1792	0.8717	0.17626
RSA2048	1.3507	0.26334

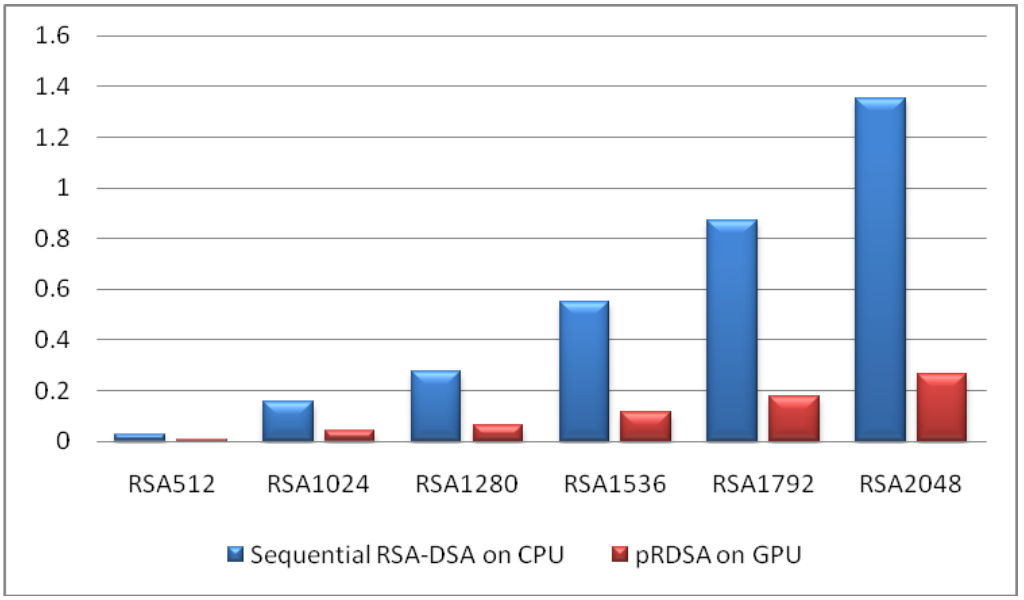

Fig 2. Performance of pRDSA on GPUs

The algorithm shows the increase in efficiency when the problem size increases keeping the number of processing elements constant. The algorithm is also cost optimal as the basic design of the algorithm has not been changed with respect to the design of serial algorithm. In addition to this the granularity for data distribution has handled efficiently to maintain pRDSA's complexity compatible to the serial algorithm.

The performance of the algorithm is affected by the time spent in data communication from host to device and vice versa. The whole process can be speedup by introducing the concept of shared memory, pinned memory and decreasing the bank conflicts occurred during the parallelization of the process. Introducing more optimization techniques can improve the results by exploiting the power of GPUs.

The algorithm shows the increase in efficiency when the problem size increases keeping the number of processing elements constant. The algorithm is also cost optimal as the basic design of the algorithm has not been changed with respect to the design of serial algorithm. In addition to this the granularity for data distribution has handled efficiently to maintain pRDSA's complexity compatible to the serial algorithm.

The performance of the algorithm is affected by the time spent in data communication from host to device and vice versa. The whole process can be speedup by introducing the concept of shared memory, pinned memory and decreasing the bank conflicts occurred during the parallelization of the process. Introducing more optimization techniques can improve the results by exploiting the power of GPUs.

7. Conclusions and Future work

In this paper, a parallel approach is proposed to generate digital signatures that can be used to provide security to the documents exchanged. SIMD model on SMP architecture is used to make optimum use of resources. The results achieved so far are satisfactory with 3-5x of speedup. However, the algorithm could be further optimized upon the synchronization issues to improve the runtime by exploiting massively parallel cores of GPUs. As pRDSA is based on SHA-1 which is a sequential algorithm to generate the hash code therefore a parallel version of hash code generation algorithm [16, 17] can also be used to speed up the whole process.

Future experiments will be based on the synchronization issues that may be faced during the data transfer between host and device. The experiments will also consider the use of pinned memory. In addition to this, further experiments of the research will also be focused on the factorization part of the RSA key generation algorithm.

References

[1] Diffie, W. & Hellman, M. Nov 1976. New directions in cryptography. Information Theory, IEEE Transactions on, 22(6), 644–654.
[2] D. Poulakis. A variant of Digital Signature Algorithm. Designs, Codes and Cryptography, 51(1):99-104, 2009.
[3] Menezes, A. J., Vanstone, S. A., & Oorschot, P. C. V. 1996. Handbook of Applied Cryptography. CRC Press, Inc., Boca Raton, FL, USA.
[4] Bewick, G. 1994. Fast multiplication algorithms and implementation.
[5] Fu, C. & Zhu, Z.-L. Oct. 2008. An efficient implementation of RSA digital signature algorithm. In Wireless Communications, Networking and Mobile Computing, 2008. WiCOM 08. 4th International Conference on, 1–4.
[6] Saxena S and Kapoor B. Parallel RSA-Based Digital Signature Algorithm For Digital Signing Applications. International Journal of Applied Engineering Research. 10(3): 7757-7764, 2015.
[7] Goldwasser, Shafi, Silvio Micali, and Ronald L. Rivest. "A digital signature scheme secure against adaptive chosen-message attacks." SIAM Journal on Computing 17.2 (1988): 281-308.
[8] Merkle, Ralph. "A certified digital signature." Advances in Cryptology—CRYPTO'89 Proceedings. Springer Berlin/Heidelberg, 1990.
[9] Li, Changxin, et al. "Efficient implementation for MD5-RC4 encryption using GPU with CUDA." Anti-counterfeiting, Security, and Identification in Communication, 2009. ASID 2009. 3rd International Conference on. IEEE, 2009.
[10] Szerwinski, Robert, and Tim Güneysu. "Exploiting the power of GPUs for asymmetric cryptography." Cryptographic Hardware and Embedded Systems–CHES 2008 (2008): 79-99.
[11] Szerwinski, Robert, and Tim Güneysu. "Exploiting the power of GPUs for asymmetric cryptography." Cryptographic Hardware and Embedded Systems–CHES 2008 (2008): 79-99.
[12] Dai, Wei, et al. "NTRU modular lattice signature scheme on CUDA GPUs." High Performance Computing & Simulation (HPCS), 2016 International Conference on. IEEE, 2016.
[13] Lin, Caiwei, Lei Zhao, and Jiwen Yang. "A cuda based implementation of an image authentication algorithm." Information Engineering and Computer Science (ICIECS), 2010 2nd International Conference on. IEEE, 2010.
[14] Saxena, Sapna, and Bhanu Kapoor. "An efficient parallel algorithm for secured data communications using RSA public key cryptography method." Advance Computing Conference (IACC), 2014 IEEE International. IEEE, 2014.

[15] Saxena, Sapna, et al. "Comparative Analysis of Sequential and Parallel Implementations of RSA." International Journal of Scientific and Engineering Research 4.8 (2013): 2100-2103.
[16] Kishore N. and Kapoor B. An efficient parallel algorithm for hash computation in security and forensics applications. Advance Computing Conference (IACC), 2014 IEEE International. IEEE, 2014.
[17] Kishore, Neha. "Parllel hashing algorithms for security and Forensic Applicatons." (2015).

Authors' Profiles

Sapna Saxena is currently working as Associate Professor in Chitkara University, Himachal Pradesh India. She received her PhD in Computer Science and Engineering from Chitkara University, Himachal Pradesh. She has many research papers and poster presentations at International Journals/Conferences in her credits. Her research interests include Network Security, Parallel Computing, Distributed Computing. She has also worked in IT industry for 8 years. She is a member of CSI and ACM.

Neha Kishore received her PhD in Computer Science and Engineering from Chitkara University, Himachal Pradesh, India in the year 2015. Her area of research includes Parallel Computing, Information Security and Digital Forensics. She is currently working as an Associate Professor in Computer Science Department, Chitkara University, H.P., India for last eight years. She has many research papers and poster presentations at International Journals/Conferences in her credits. She is also a member of ACM, IAENG, Internet Society, CSI and has been also certified as an ACM Ambassador.

How to cite this paper: Sapna Saxena, Neha Kishore," PRDSA: Effective Parallel Digital Signature Algorithm for GPUs ", International Journal of Wireless and Microwave Technologies(IJWMT), Vol.7, No.5, pp. 14-21, 2017.DOI: 10.5815/ijwmt.2017.05.02