# Comparison between RSA Hardware and Software Implementation for WSNs Security Schemes

Abdullah Said Alkalbani, Teddy Mantoro, Abu Osman Md Tap
Department of Computer Science, Kulliyyah (Faculty) of Information & Communication Technology
International Islamic University Malaysia, Kuala Lumpur, Malaysia

***Abstract*— The need of security to protect the data through networks has become of vital importance and critical for many sensor network applications. There are several security schemes implemented using hardware or software trying to solve the problem of security in WSN by taking into consideration the limitations of sensors (bandwidth and energy), the majority of them are symmetric key encryption schemes and some others are asymmetric encryption schemes is not recommended to be used because of high time complexity and consumption demand. In this study we compare the time complexity and power consumption between software and hardware implementation using RSA algorithm. Our simulation shows that usage of hardware security could improve time efficiency and decrease the power consumption, so the strong cryptography can be implemented in WSNs security.**

**Index Terms—AES, assymetric encryption, DES, FPGA, key management, symetric encryption.**

## I. Introduction

RECENT advances in electronics and wireless communication technologies have enabled the development of large-scale wireless sensor networks that consist of many low-power, low-cost, and small-size sensor nodes. Sensor networks hold the promise of facilitating large-scale and real-time data processing in complex environments. It has emerged as a new monitoring and control solution for various ubiquitous applications [1]. The need of security to protect the data through networks has become of vital importance and critical for many sensor network applications, such as military target tracking and security monitoring. This area of research has become quite active in the recent years.

Encryption is mainly used to ensure secrecy of data. It can also be used to secure authentication, certainty about the identity of the sender, and integrity, certainty about the unimpaired ness of data. This is especially important if electronic data-flow is to have legal consequences. Encryption algorithms have received wide attention and study. Encryption algorithms can be classified into two groups: asymmetric encryption algorithms (with public key algorithms) such as Data Encryption Standards (DES) and symmetric encryption algorithms (with private Key algorithms). Most encryption algorithms are implemented at software implementations.

Among software and hardware implementations for WSN security selecting suitable implementation is critical issue due to the constraints on WSNs. The problem in WSNs security is that the public key cryptography algorithm is not recommended to be used because of high time complexity and consumption demand.

Through this study, we compare between software and hardware implementation in terms of time complexity and power consumption and proposed to be used in WSNs by using public key cryptography algorithm, such as RSA algorithm.

The advantages of software implementations are ease of use, ease of upgrading, portability and flexibility. However a hardware implementation has more physical security by nature, as it can not easily be modified by an attacker [2]. To provide security and privacy to small sensor nodes is challenging, due to the limited capabilities of sensor nodes in terms of computation, communication, memory/storage, and energy supply [3].

Every hardware circuit can be characterized by two major parameters: speed of operation, and area. Cryptographic algorithms are intended to perform cryptographic transformations on strings of data. Therefore the speed of cryptographic implementations is commonly characterized by the throughput. Throughput does not always give the full information about the speed, and is often accompanied by another parameter called latency [4].

The main contribution of this work is to prove that hardware implementation for WSNs security, even using strong cryptography, is much more effective in time and power consumption than software solutions.

The paper is organized as follows. In Section II, related work in WSNs security implementations discussed. In section III, the security requirements in WSNs are described. Hardware and software implementations for RSA algorithm are discussed in Section IV and Section V respectively. In Section VI, the results of comparison between hardware and software implementations are briefly described. We conclude this paper in Section VII.

## II. Related Work

Security is a very important issue when designing or deploying any network or protocol. However the recently developed networks as the wireless have not given the necessary attention to security when designing protocols by

taking into account the specificity of these networks as the used medium and the devices constraints [5]. Thus, many security protocols were proposed trying to efficiently carry out the problem of security and the constraints of wireless networks [6]. However, in sensor network, the problem of security is more challenging regarding the limitation of sensors and the area where the sensors are deployed such as battlefields [7].

WSNs have unique constraints as compared to traditional networks making the implementation of existing security measures not practicable. These constraints are the result of WSNs limitations which make the design of security procedures more complicated.

Currently, WSNs security has many challenges. One challenge is how to improve the security of data transmission between WSN nodes against eavesdropping, tampering and modification of packets. The other challenge is the need of secure and efficient key-distribution mechanism in allowing simple key establishment for large-scale sensor networks to be used for security protocols. Another challenge lies in the needs to balance data integrity, confidentiality, and availability as well as preserving constraint energy resources.

The severe constraints and demanding environments of WSN make computer security for these systems even more challenging. Its required to to choose the best security implementation method software or hardware to fit the security demands.

## III. Security Requirements In WSNs

All security mechanisms require a certain amount of resources for the implementation such as: data memory, code space and energy to power the sensor. These resources, however, are very limited in a tiny wireless sensor:

a. Limited Memory and Storage Space.
b. Power Limitation.
c. Vulnerability of nodes to physical capture.
d. Lack of a-priori knowledge of post-deployment configuration.
e. Collisions and latency.

Wireless Sensor Networks are vulnerable to many attacks because of broadcast nature of transmission medium, resource limitation on sensor nodes and uncontrolled environment [8]. The security requirements of wireless sensor networks are [9]:

a. Confidentiality.
b. Integrity.
c. Authentication.

Through this study, the strong cryptography algorithm, such as RSA algorithm is proposed to be used in WSNs even there a lot of constrains including high time complexity and power consumption. The RSA coding algorithms are outlined the four processes needed for RSA encryption, i.e.:

a. Creating public key
b. Creating a private (secret) key
c. Encrypting messages
d. Decoding message

In brief, the RSA algorithm is the following [16]:

To create public key $K_p$:

a. Select two different primes $P$ and $Q$
b. Assign $x = (P-1)(Q-1)$
c. Choose $E$ relative primes to $x$, which must satisfy a condition for $K_s$.
d. Assign $N = P*Q$
e. $K_p$ is $N$ concatenated with $E$.

To create private key:

a. Choose $D$: $D*E \bmod x = 1$
b. $K_s$ is $N$ concatenated with $D$.

To encode plain text m by :

a. Assume $m$ is a numeric
b. Calculate $c=m^E \bmod N$.

To decode c back to m:

a. Calculate $m = c^D \bmod N$.

## IV. Hardware Implementation

Until very recently, all encryption products were in the form of specialized hardware. These encryption/decryption boxes plugged into a communications line and encrypted all the data going across that line. Although software encryption is becoming more prevalent today, hardware is still the embodiment of choice for military and serious commercial applications.

The NSA, for example, only authorizes encryption in hardware. There are several reasons why this is so. The first is speed. Encryption algorithms consist of many complicated operations on plaintext bits. These are not the sorts of operations that are built into your run-of-the-mill computer. The two most common encryption algorithms, DES and RSA, run inefficiently on general-purpose processors. While some cryptographers have tried to make their algorithms more suitable for software implementation, specialized hardware will always win a speed race. Additionally, encryption is often a computation-intensive task. Tying up the computer's primary processor for this is inefficient. Moving encryption to another chip, even if that chip is just another processor, makes the whole system faster.

The second reason is security. An encryption algorithm running on a generalized computer has no physical protection. Mallory can go in with various debugging tools and surreptitiously modify the algorithm without anyone ever realizing it. Hardware encryption devices can be securely encapsulated to prevent this. Tamperproof boxes can prevent someone from modifying a hardware encryption device. Special-purpose VLSI chips can be coated with a chemical such that any attempt to access their interior will result in the destruction of the chip's logic. IBM developed a cryptographic system for encrypting data and communications on mainframe computers. It includes tamper resistant modules to hold keys.

The final reason for the prevalence of hardware is ease of installation. Most encryption applications don't involve general-purpose computers [4]. People may wish to encrypt

their telephone conversations, facsimile transmissions, or data links. It is cheaper to put special-purpose encryption hardware in the telephones, facsimile machines, and modems than it is to put in a microprocessor and software. Even when the encrypted data comes from a computer, it is better to install a dedicated hardware encryption device than to modify the computer's system software. Encryption should be invisible; it should not hamper the user. The only way to do this in software is to write encryption deep into the operating system. This isn't easy. On the other hand, even a computer neophyte can plug an encryption box between his computer and his external modem.

The three basic kinds of encryption hardware on the market today are: self-contained encryption modules (that perform functions such as password verification and key management for banks), dedicated encryption boxes for communications links, and boards that plug into personal computers. Some encryption boxes are designed for certain types of communications links, such as T-1 encryption boxes that are designed not to encrypt synchronization bits. There are different boxes for synchronous and asynchronous communications lines. Newer boxes tend to accept higher bit rates and are more versatile. Even so, many of these devices have some incompatibilities. Buyers should be aware of this and be well-versed in their particular needs, lest they find themselves the owners of encryption equipment unable to perform the task at hand. Pay attention to restrictions in hardware type, operating system, applications software, network, and so forth.

PC-board encryptors usually encrypt everything written to the hard disk and can be configured to encrypt everything sent to the floppy disk and serial port as well. These boards are not shielded against electromagnetic radiation or physical interference, since there would be no benefit in protecting the boards if the computer remained unaffected. More companies are starting to put encryption hardware into their communications equipment. Secure telephones, facsimile machines, and modems are all available. Internal key management for these devices is generally secure, although there are as many different schemes as there are equipment vendors. Some schemes are more suited for one situation than another, and buyers should know what kind of key management is incorporated into the encryption box and what they are expected to provide.

Security of WSNs can be improved by using cryptographic hardware through the following:

- Sensor nodes can be improved by an add-on secure microcontroller with secure memory and hardware crypto engine supporting AES and 3DES encryption standards. The encryption engine increases encrypted communication speed up several Mbps. Unfortunately, current crypto-accelerators suitable for sensor networks have no support for public key cryptography.
- Sensor node can be extended with an FPGA module implementing fast symmetric and asymmetric cryptographic algorithms. This solution offers high-speed encryption and key generation at highest price and power consumption (depends on FPGA module).
- Smart cards, as tamper resistant devices, can be used as crypto accelerators and also as a secure storage for cryptographic keys. These crypto-accelerators ordinarily support symmetric and asymmetric cryptography together with low power consumption [10].

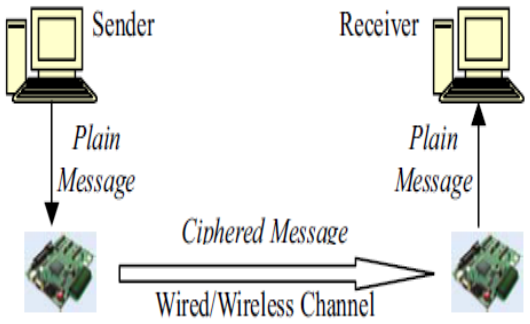


Fig. 1. Schematic diagram of the WSNs hardware security system.

In general the security implementation process could be represented as a communication from sending-end to receiving-end as shown in Figure 1. In WSNs it can be seen that a Sender node stores a Plain Message to be sent to Receiver base station. The encryption processor encrypts the Plain Message to Ciphered Message to protect the eavesdropping before sending it off. When the Ciphered Message arrives at receiving-end, its decryption processor executes the decryption process to recover the Plain Message for the Receiver base station.

An important benefit of using hardware implementation is *tamper-resistant* hardware. It is a concept of hardware that is resistant against physical attacks. Secure hardware is equipped by microprocessor and secures memory, which contains sensitive data and algorithms for processing these data [11]. The data never leave the secure memory and cannot be modified by another code. Such secure hardware cannot be cloned or emulated, because these features cannot be replaced just by software without special hardware support. Most widely used kind of tamper-resistant device is smart card.

## V. Software Implementation

Any encryption algorithm can be implemented in software. The disadvantages are in speed, cost, and ease of modification (or manipulation). The advantages are in flexibility and portability, ease of use, and ease of upgrade. The algorithms can be inexpensively copied and installed on many machines. They can be incorporated into larger applications, such as communications programs or word processors. Software encryption programs are popular and are available for all major operating systems. These are meant to protect individual files; the user generally has to manually encrypt and decrypt specific files. It is important that the key management scheme be secure: The keys should not be stored

on disk anywhere (or even written to a place in memory from where the processor swaps out to disk). Keys and unencrypted files should be erased after encryption. Many programs are sloppy in this regard, and a user has to choose carefully.

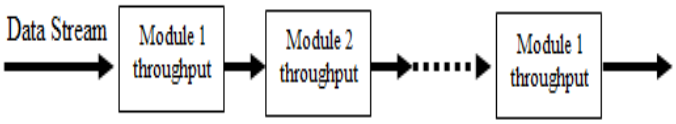


Fig. 2. system consisting of multiple modules with throughput parameters

There are many factors effects on the software implementation. The most important factors are throughput, bandwidth and latency.

Throughput is defined as the number of bits processed in a unit of time after the process has gone through any initialization, and is usually expressed in Mbps or Gbps.

*Throughput = Number of processed bits / Time unit – Startup*

Typically, the encryption and decryption throughputs are equal. All symmetric-key algorithms perform a fixed sequence of transformations. In other words, no conditional operations are performed. Therefore, the time of encryption of one block of data is usually fixed and known, unless implementation uses some tricks which can vary the time of encryption. From the point of view of cryptographers, using any techniques yielding correlation between data and time of encryption is highly undesirable. Such a correlation leaks information about data, and can be used to mount timing attacks on the implementation.

In security, bandwidth is defined as the number of cipher blocks encrypted or decrypted per second. The unit of bandwidth is 1/s. The equation of throughput can described as

*Throughput = block size * Bandwidth*

Throughput has a very important meaning when considering a bigger system consisting of multiple modules processing data in sequence, as shown in Figure 2 below, because it is limited to the maximum throughput of the slowest of the modules. Cryptographic transformations usually require the most processing power, and present a bottleneck in many applications [4].

The third important factor in software implementation is *latency*. It is defined as the time required to complete processing one block of data, and is usually expressed in number of cycles. This is the time between a moment when a block of data enters the encryption unit, and a moment when it leaves it. The unit of latency is ns (nanosecond). The encryption latency and throughput are related by simultaneously/latency).

The total latency of a system is a sum of latencies of all modules processing data sequentially. Therefore, all modules, no matter how different from each other, contribute to the overall latency.

## VI. Results and Discussion

Cryptographic algorithms are utilized for security services in various environments in which low cost and low power consumption are key requirements. Wireless Local Area Networks (WLAN), Wireless Personal Area Networks (WPAN), Wireless Sensor Networks (WSN), are examples of such technologies. A solution to reduce power and memory costs, as well as greatly increasing speed of algorithm, is to use a hardware implementation of a cipher [13]. Compared to software, significantly higher performance and lower power consumption can be achieved with dedicated hardware.

In this work we focused on public key cryptography, mainly in RSA cryptosystem, as secure but also very computational complex algorithm. Previous works showed that RSA is not suitable for WSN because of high time complexity and consumption demands [12]. Our simulation showed that usage of hardware could improve not only time efficiency and security but also decrease power consumption of sensor nodes in case of using strong cryptography.

TABLE 1
Comparison of time complexity and power consumption using software and hardware implementation for rsa algorithm

| Implementation | Key length | Time(s) | Power (mWs) |
|---|---|---|---|
| RSA software | 1024 | 22.03 | 726.99 |
| RSA software | 2048 | 166.85 | 5506.05 |
| RSA hardware | 1024 | 0.75 | 27.15 |
| RSA hardware | 2048 | 1.89 | 79.09 |

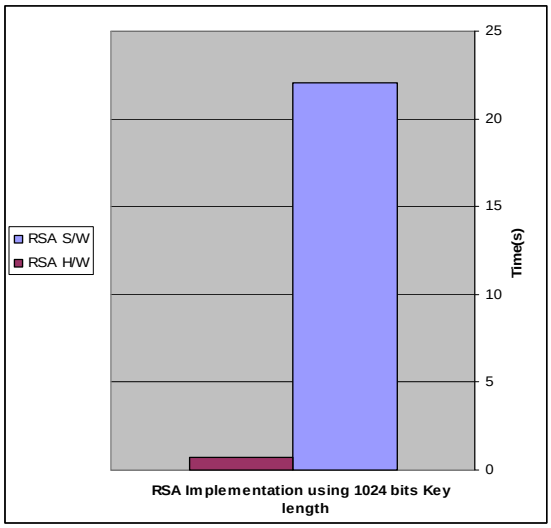


Fig. 3. Comparison of time complexity using 1024 bits key length software and hardware implementation for RSA algorithm.

Evaluations of security architectures are usually based on analytical modeling and performance evaluations are usually based on simulation models.

Finally we compared power consumption and time complexity hardware and software implementation of RSA. We prove that hardware platforms are much more effective in time and power consumption than software solutions as shown in table 1.

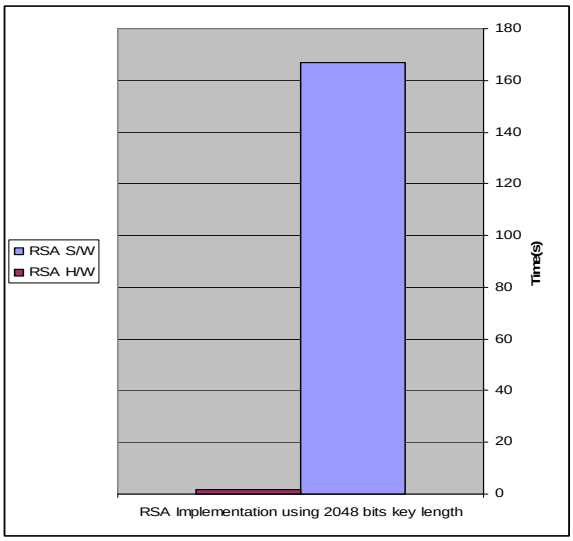


Fig. 4. Comparison of time complexity using 2048 bits key length software and hardware implementation for RSA algorithm.

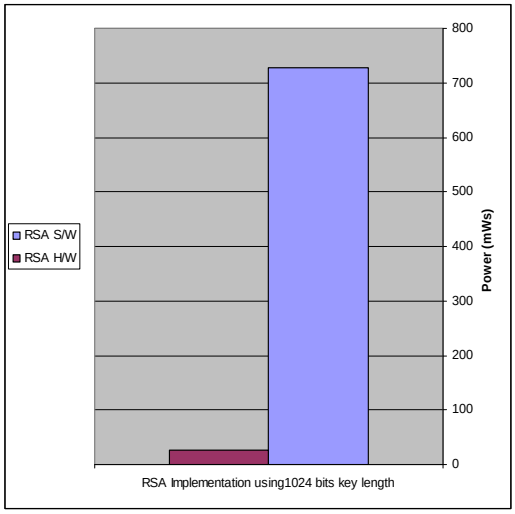


Fig. 5. Comparison of power consumption using 1024 bits key length software and hardware implementation for RSA algorithm.

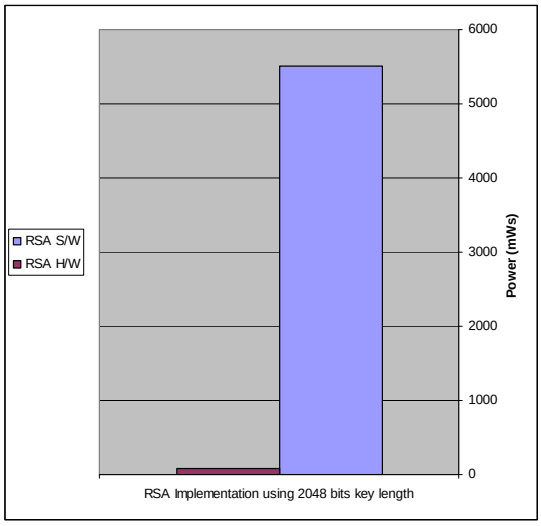


Fig. 6. Comparison of power consumption using 2048 bits key length software and hardware implementation for RSA algorithm.

The results in Table 1 and Figures 3-6 show that time complexity and power consumption can be reduced up to 30 times for RSA-1024. When 2048-bit keys are used, time demands can be reduced up to 88 times and power consumption up to 70 times if hardware implementation used.

## VII. CONCLUSION

Many issues still remain to be addressed in respect to achieving security on WSNs. Sensor nodes are not usually tamper resistant, because of hardware limitation and cost issues [14, 15]. Hence, attackers can get sensor nodes, extract stored keys, and then can insert malicious code. This node compromising is one of the main issues in wireless sensor networks. Second, sensor nodes are limited in their energy and computation abilities. Because of that, there exists a fundamental necessity of obtaining more knowledge about suitable implementation method for security schemes in factor of performance and low power consumption.

As in the literature showed that RSA is not suitable for WSN because of high time complexity and consumption demand [12]. We reviewed hardware and software architectures proposed WSNs security implementations by presented the comparison between hardware and software implementations public key cryptography algorithm, such as RSA algorithm. Our simulation showed that usage of hardware could improve not only time efficiency and security but also decrease power consumption of sensor nodes in case of using strong cryptography.

This also proves that hardware architecture for security of WSNs is more efficient than software implementation in terms of time complexity, throughput and power consumed during the security process.

This will allow us to identify further security mechanisms that can be successfully employed on WSNs, and also gain more knowledge towards the definition of a complete security architecture for WSNs.

## REFERENCES

[1] S. M. Hwang, and E. N. Huh. "An Efficient Topology Control and Dynamic Interval Scheduling Scheme for 6LoWPAN", in Gervasi et al. (Eds.): ICCSA, Part I, LNCS 5592, pp. 841–852. 2009

[2] A. AlKalbany, M. Saeb, and H. A. AlHassan. "FPGA Implementation of the Pyramids Block Cipher", System on Chip Conference (SOCC2005), Washington, US.2005

[3] J. Albath, and S. Madria. "Practical Algorithm for Data Security (PADS) in Wireless Sensor Networks". *MobiDE'07, , Beijing, China.2007*

[4] K. Gaj, and P. Chodowiec. "Comparison of the hardware performance of the AES candidates using reconfigurable hardware", Proc. 3 rd Advanced Encryption Standard (AES) Candidate Conference, New York.2000

[5] R. Ramanathan, and J. Redi. "A Brief Overview of Ad Hoc Networks: Challenges and Directions", *IEEE Communication Magazine*, vol. 40, no. 5, pp. 20-22.2002.

[6] B. Kadri, A. Mhamed, and M. Feham "Secured Clustering Algorithm For Mobile Ad Hoc Networks", *International Journal of Computer Science and Network Security*, Vol.7, No.3, pp 27-34.2007.

[7] A. Perrig, J. Stankovic, and D. Wagner, "Security In Wireless Sensor Networks", *Communications of the ACM*, vol. 47, no. 6, pp. 53--57. 2004.

[8] L. Zhijun, and G. Guang, "Security In Wireless Sensor Networks", University of Waterloo, CACR 2008-20.

[9] A. Perrig,, J. Stankovic, and D. Wagner, "Security In Wireless Sensor Networks", *ACM*,Vol. 47, No. 653.2004.

[10] P. Pecho, J. Nagy, and P. Hanacek, "Power Consumption Of Hardware Cryptography Platform For Wireless Sensor", International Conference on Parallel and Distributed Computing, Applications and Technologies, 2009.

[11] J. P. Anderson. "Computer Security Technology Planning Study", ESD-TR-73-51, vol. I. ESD/AFSC, Hanscom AFB, Bedford, Mass., (NTIS AD-758 206), 1972.

[12] F. Amin, A. H. Jahangir, H. Rasifard, "Analysis Of Publickey Cryptography For Wireless Sensor Networks Security", In Proceedings of World Academy of Science, Engineering and Technology, ISSN 1307-6884 , 2008.

[13] M .Healy, T. Newe, and E. Lewis. "*Analysis Of Hardware Encryption Versus Software Encryption On Wireless Sensor Network Motes*"., Springer-Verlag Berlin Heidelberg 2008.

[14] A. Perrig, R. Szewczyk, V.Wen, D. E. Culler, and J. D. "Tygar, Spins: security protocols for sensor networks", MOBICOM, pp. 189–199, 2001.

[15] S. Setia, and S. Jajodia. "Leap: Efficient Security Mechanisms For Large-Scale Distributed Sensor Networks", CCS '03: Proceedings of the 10th ACM conference on Computer and communications security (New York, NY, USA), ACM Press, pp. 62–72, 2003.

[16] H. Anderson. *Introduction to Computer Security*, Prentice Hall, 2004, pp: 85-86.