International Journal of Information, Control and Computer Sciences
ISSN: 2517-9942
Vol:2, No:5, 2008

Analysis of Public-Key Cryptography for Wireless Sensor Networks Security

F. Amin, A. H. Jahangir, and H. Rasifard

***Abstract*—**With the widespread growth of applications of Wireless Sensor Networks (WSNs), the need for reliable security mechanisms these networks has increased manifold. Many security solutions have been proposed in the domain of WSN so far. These solutions are usually based on well-known cryptographic algorithms.

In this paper, we have made an effort to survey well known security issues in WSNs and study the behavior of WSN nodes that perform public key cryptographic operations. We evaluate time and power consumption of public key cryptography algorithm for signature and key management by simulation.

***Keywords*—**Wireless Sensor Networks, Security, Public Key Cryptography, Key Management.

I. Introduction

WIRELESS sensor networks consist of small nodes that sense their environment, process data, and communicate through wireless links. They are expected to support a wide variety of applications, many of which have at least some requirements for security.

Cryptographic algorithm for authentication and encryption can be implemented in two ways: using public keys or private keys. When using public keys, the key value of every node is public information, and is therefore known by all other nodes. When a node wants to communicate privately with another node, the source node simply encrypts data using the public key of the sink node. In this case, only the sink node can correctly decrypt the data. This method is called asymmetric key encryption because the two communicating nodes use different keys during the session. When using private keys, nodes must first agree on a key before they can communicate securely. One possibility is to use public keys to encrypt data from which private keys can be derived. Private Key algorithms are based on symmetric key encryption because both communicating nodes use the same keys for encrypting and decrypting data.

In wired data networks, nodes rely on pre-deployed trusted server to help establish trust relationships but in WSN, these trusted authorities do not exist because sensor nodes have limited memory, CPU power, and energy, hence cryptographic algorithms must be selected carefully.

A survey of security issues in ad hoc and sensor networks can be found in [1]. Related work in the security area, focused on WSN, is summarized in [2,3].

All approaches for enabling security in WSNs are very scenario dependent. There exist different requirements, for example, in an agriculture application [4] compared an habitat monitoring scenario [5]. Other requirements appear in the operation and control domain. Sensor nodes must be reconfigured, calibrated, and reprogrammed [6]. Such operations are very sensible to possible attacks. Finally, it must be mentioned that they ignore the problem of key management. Several solutions that address this issue have been proposed so far (see for example, [7]).

In Section II we will survey the security requirements in WSNs. Section III and IV deal with Public key cryptosystem and Key management issues in WSNs, respectively. Section V presents an evaluation of energy consumption in public key cryptosystem. Section VI concludes the paper and proposes some future work.

Authors are with Sharif University, Tehran, Iran (e-mail: f_amin@ce.sharif.ac.ir, jahangir@sharif.ac.ir, rasifard@ce.sharif.ac.ir).

II. Security Requirements

The security of WSNs can be classified into two broad categories: operational security and information security. The operation-related security objective is that a network, as a whole, should continue to function even when some of its components are attacked (the *service availability* requirement). The information-related security objectives are that *confidential* information should never be disclosed, and the *integrity* and *authenticity* of information should always be assured. These objectives are marked with a cross in Table I if they are violated in the corresponding scenario [9].

TABLE I
Potential Security Threats, Grouped According to Application Domains [9]

Application domain	Potential security threats	Properties violated			
		S	C	I	A
Military	Denial-of-service attacks	×		×	
	Eavesdropping of classified information		×		
	Supply of misleading information				×
Disaster Detection	Supply of misleading information				×
Environmental Monitoring	Suppose government-endorsed environmental sensors are installed near a factory to monitor air/water quality to make sure the factory's emission lies beneath the pollution threshold, however by feeding the sensors with wrong information, the factory allows itself to escape detection and let its polluting emission go unchecked.			×	
Intelligent Buildings	Biometrics-based access control mechanisms are compromisable if the biometric sensors can be bypassed or fooled.	×		×	
	Token-based access control mechanisms are compromisable if the token authentication protocol				×

	is insecure.				
Health/Medical	Providing wrong physiological measurements of a patient to the career or doctor, a miscreant may cause potentially fatal diagnosis and treatment to be performed on the patient.			×	×
Transportation	There is no order in the city when traffic information can no longer be trusted because they can easily be spoofed.			×	×
Industry	Eavesdropping of commercial secrets by business rivals.		×		
	Intentional disruption of manufacturing processes as a result of misleading sensor readings caused by disgruntled employees or business spies.	×		×	
Space exploration	Space agencies invest billions into space exploration projects, it is only logical that they want to ensure all commands executed on their space probes are authorized, and all collected data encrypted and authenticated.	×	×	×	×

S=Service Availability, **C**=Confidentiality, **I**=Integrity, **A**=Authenticity

While it may seem that information security can readily be achieved with cryptography, there are two facts that make achieving the above objectives non-trivial in WSNs:

- As sensor nodes operate unattended they are potentially accessible, both geographically and physically, to any malicious party imaginable;
- Sensor nodes communicate through an open medium.

A. Confidentiality

Confidentiality means keeping information secret from unauthorized parties. A sensor network should not leak sensor readings to neighboring networks.

The confidentiality objective is required in sensors' environment to protect information traveling between the sensor nodes of the network or between the sensors and the base station from disclosure, since an adversary having the appropriate equipment may eavesdrop on the communication. By eavesdropping, the adversary could overhear critical information such as sensing data and routing information.

B. Authentication

In a sensor network, an adversary can easily inject messages, so the receiver needs to make sure that the data used in any decision-making process originates from the correct source.

As in conventional systems, authentication techniques verify the identity of the participants in a communication, distinguishing in this way legitimate users from intruders. In the case of sensor networks, it is essential for each sensor node and base station to have the ability to verify that the data received was really sent by a trusted sender and not by an adversary that tricked legitimate nodes into accepting false data. If such a case happens and false data are supplied into the network, then its behavior could not be predicted, and most of times the mission of WSN will not be accomplished as expected.

However, authentication for broadcast messages requires stronger trust assumptions on the network nodes. The creators of SPINS [10] contend that if one sender wants to send authentic data to mutually untrusted receivers, using a symmetric MAC is insecure since any one of the receivers know the MAC key, and hence could impersonate the sender and forge messages to other receivers. LEAP [11] uses a globally shared symmetric key for broadcast messages to the whole group.

C. Integrity

Data integrity ensures the receiver that the received data is not altered in transit by an adversary. Lack of integrity could result in many problems since the consequences of using inaccurate information could be disastrous, for example, for the healthcare sector where lives are endangered. Integrity controls must be implemented to ensure that information is not altered in any unexpected way.

D. Freshness

One of the many attacks launched against sensor networks is the message replay attack where an adversary may capture messages exchanged between nodes and replay them later to cause confusion to the network. Data freshness implies that the data is recent, and it ensures that an adversary has not replayed old messages.

To achieve freshness, network protocols must be designed in a way to identify duplicate packets and discard them preventing potential mix-up.

E. Availability

Availability ensures that services and information can be accessed at the time they are required. In sensor networks there are many risks that could result in loss of availability such as sensor node capturing and denial of service attacks.

The availability of a sensor and sensor network may decrease for the following reasons [12]:

- Additional computation consumes additional energy. If no more energy exists, the data will no longer be available.
- Additional communication also consumes more energy.Besides, as communication power increases so does the chance of a communication conflict or interference.
- A single point failure exists if we use the central point scheme such as a single sink or gateway. This greatly threatens the availability of the network.

F. Secure Management

Management is required in every system that is constituted of multi components, and handles sensitive information. In the case of sensor networks, we need secure management on base station level; since sensor nodes communication ends up at the base station, issues like key distribution to sensor nodes in order to establish encryption and routing information need secure management.

G. Quality of Service

Assuring the quality of Service objective is a big challenge to security designers. As sensor networks have several limitations (e.g. energy, processing and memory capacities etc.), the achievement of quality of service

becomes even more constrained. Security mechanisms must be lightweight so that the overhead caused, for example, by encryption be minimized and do not affect the performance of the network. Performance and quality in sensor networks involve the timely delivery of data to prevent the loss of critical data or events, and the accuracy with which the data are reported compared to what is actually occurring in their environment [13].

III. Public-Key Cryptosystem

A public key cryptosystem employs a pair of different but associated keys. One of these keys is released to the public while the other, the private key, is known only to its owner. It is designed to be computationally intractable to calculate a private key from its associated public key; In other words, it is believed that any attempt to compute this key will fail during the lifetime of the network or the duration of an operation even when up-to-date technology and equipment are used.

With a public key cryptosystem, the sender can encrypt a message using the receiver's public key without needing to know the private key of the receiver. Therefore, they are suitable for communication among the general public.

Today, three types of systems, classified according to the mathematical problem on which they are based, are generally considered both secure and efficient. They are classified as follows:

- The *integer factorization* systems.
- The *discrete logarithm* systems.
- The *elliptic curve discrete logarithm* systems.

A. Public-Key Encryption (PKE)

A PKE is a triple of PPT algorithms $E = (G; E; D)$ where:

a) G is the key-generation algorithm. $G(1^k)$ outputs $(PK; SK; M_k)$, where SK is the Secret key, PK is the public-key, and M_k is the message space associated with the PK/SK-pair. Here k is an integer usually called the security parameter, which determines the security level.

b) E is the encryption algorithm. For any $m \in M_k$, E outputs $c \xleftarrow{r} E(m;PK)$ the encryption of m. c is called the cipher text. We sometimes also write $E(m;PK)$ as $E_{PK}(m)$, or $E(m;r;PK)$ and $E_{PK}(m;r)$, when we want to emphasize the randomness r used by E.

c) D is the decryption algorithm. $D(c;SK) \xrightarrow{r} \tilde{m} \in \{invalid\} \cup M$ is called the decrypted message. We also sometimes denote $D(c;SK)$ as $D_{SK}(c)$ and remark that usually D is deterministic.

d) We require the correctness property, i.e. everybody behaves as assumed:
$\forall m \in M_k,\ \tilde{m} = m,\ that\ is\ D_{SK}(E_{PK}(m)) = m$

Let us check that RSA satisfies the above definition. Notice, both E and D are deterministic.

a) $G(1^k)$ corresponds to the following algorithm: p *and* q are random primes of k bits, $n=pq$; $e \xleftarrow{r} Z^*_{\varphi(n)}$, $d = e^{-1} \bmod \varphi(n)$, $M_k = Z^*_n$. Set $PK= (n, e)$, $SK=d$.

b) $c = E(m;(n,e)) = m^e \bmod n.$

c) $\tilde{m} = D(c;(d,n)) = c^d \bmod n.$

More generally, we could construct a PKE from any TDP. Suppose we have a TDP $\Re$ with trap-door information t_k and algorithm I for inversion. Here is the induced PKE:

a) $G(1^k) \xrightarrow{r} (\Re, t_k, \{0,1\}^k)$, and $\Re$ is the PK and trapdoor t_k is the SK.

b) $E(m;PK) = \Re(m).$

c) $D(m;SK) = I(c,t_k).$

B. RSA Cryptosystem

The RSA-system is based on the difficulty of factoring, $P = C = Z/nZ$ for an integer $n = p.q$, where n (the modulus) is known to everybody, but the prime factors p, q are known only to receiver. We need in practice p and q to be very large. We take K to be the set of positive integers relatively prime to $lcm(p-1,q-1)$. The encryption key $e \in K$ is known to everyone, but the decryption key $d \in K$ is known only to receiver. Then sender encrypts:

$$E : P \times K \longrightarrow C, \quad E(a,e) = a^e \quad mod\ n$$

To decrypt the cipher text, receiver:

$$D : C \times K \longrightarrow P, \quad D(b,e) = b^d \quad mod\ n$$

where $ed \equiv 1 (mod\ lcm(p-1,q-1))$.

C. Elliptic Curve Cryptosystem (ECC)

Elliptic curves are an algebraic structure, and their use for cryptography was first mentioned in [14] and [15]. They feature properties which allow the setup of a problem similar to the well known discrete logarithm problem of finite fields – also known as Galois fields (GF).

The subsequent section gives a brief and rough mathematical background to understand our implementation.

In recent years, ECC has attracted much attention as the security solutions for wireless networks due to the small key size and low computational overhead.

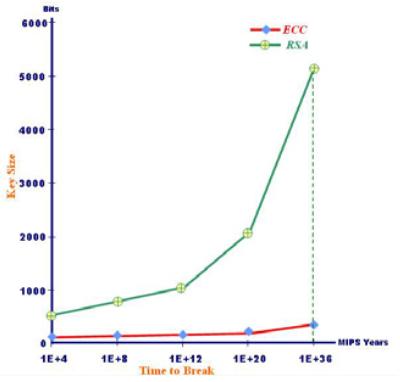

Fig. 1 Comparison of Security Levels

Fig. 1 compares the time required to break ECC with the time required to break RSA for various modules sizes and using the best general algorithms known. The running times are computed in MIPS years.

From Fig. 1, we see that to achieve reasonable security, RSA should employ 1024-bit module, while a 160-bit module should be sufficient for ECC. Moreover, the security gap between the systems increases dramatically as the module sizes increases. For example, 160-bit ECC offers the comparable security to 1024-bit RSA.

ECC includes key agreement, encryption, and digital signature algorithms. The key distribution algorithm is used to share a secret key, the encryption algorithm enables confidential communication, and the digital signature algorithm is used to authenticate the signer and validate the integrity of the message:

Key Agreement:

- ECMQV: Elliptic Curve Menezes-Qu-Vanstone
- ECDH: Elliptic Curve Diffie-Hellman

Encryption:

- ECIES: Elliptic Curve Integrated Encryption Standard

Digital Signatures:

- ECDSA: Elliptic Curve Digital Signature Algorithm
- ECPVS: Elliptic Curve Pintsov Vanstone Signatures
- ECNR: Elliptic Curve Nyberg Rueppel

Let *K* be a field. For example, *K* can be the finite (extension) field F_{q^r} of F_q, the prime field Z_p where *p* is a large prime, the field *R* of real numbers, the field *Q* of rational numbers, or the field *C* of complex numbers.

An elliptic curve over a field *K* is defined by the *Weierstrass* equation:

$$y^2 + a_1xy + a_3y = x^3 + a_2x^2 + a_4x + a_6$$

Where $a_1, a_3, a_2, a_4, a_6 \in K$ and a_i are the parameters of the curve.

The elliptic curve *E* over *K* is denoted *E(K)*. The number of points on *E* is denoted *#E(K)* or just *#E*.

Since the field F_q is generally used in cryptographic applications, *Weierstrass* equation can be simplified to:

$$y^2 = x^3 + ax + b$$

Where $a, b \in F_q$ and $4a^3 + 27b^2 \neq 0$, together with a special point ∞ called the point at infinity.

IV. Key Management

Key management is a fundamental security issue in sensor networks. It is the basis to establish the secure communication using cryptographic technologies between sensor nodes in a sensed area.

Key management is the process by which cryptographic keys are generated, stored, protected, transferred, loaded, used, and destroyed. There are four principal concerns in a key management framework:

- How many keys are needed and how should the keys be distributed before the nodes are deployed? This is a problem of *key deployment*/*pre-distribution*.
- How does any pair of nodes or a group of nodes establish a secure session? This is a problem of *key establishment*.
- How should a node be added to the network such that it be able to establish secure sessions with existing nodes in the network, while not being able to decipher past traffic in the network? This is a problem of *member*/*node addition*.
- How should a node be evicted from the network such that it will not again be able to establish secure sessions with any of the existing nodes in the network, and not be able to decipher future traffic in the network? This is a problem of *member*/*node eviction*.

The major advantages and drawbacks of different key distribution and management schemes are summarized in Table II.

A. Key Establishment Protocols

Key establishment in sensor networks can also be realized with protocols where the nodes set up a shared secret key after deployment, either through key transport or key agreement [16]. A key transport protocol is a protocol where one entity creates or otherwise obtains a secret key and transfers it securely to the other entity. Key agreement refers to a mechanism or protocol where all participating entities contribute a random input which is used to derive a shared secret key. The advantage of key agreement over key transport is that no entity can predetermine the resulting key as it depends on the input of all participants.

TABLE II
THE MAJOR KEY MANAGEMENT SCHEMES

Key Management Schemes	Advantage	Drawback
Network Wide Shared Key	One of the simplest schemes, in which a single network wide symmetric key is used by every node.	An adversary can extract the network wide shared key by capturing a single node.
Master Key and Link Keys	Every node is preconfigured with a master key; every node fetches a set of link keys corresponding to its each communication link with other nodes.	-This is resilient to a single node compromise attack. -Addition of new nodes is not possible. -The link keys cannot be securely transmitted over the network.
Public Key Cryptography	Very good solution for key management and distribution in traditional WSN.	The memory and processing constraints of these tiny devices rule out the possibility for using this scheme.
Preconfigured Symmetric Keys	Every node in the network is preconfigured with a set of link keys with which it will establish secure links with other nodes.	This is not scalable as every node has to store *n(n - 1)/2* keys, if *n* is the number of nodes in the network.
Bootstrapping Keys	Allows an on-demand key generation for a secure connection established between the nodes.	Suffers from single point of failure as base station has to maintain a database for the link keys.

Key agreement problem is a part of the *key management* problem, which has been widely studied in general network

environments. There are three types of general key agreement schemes: trusted-server scheme, self-enforcing scheme, and key pre-distribution scheme [17].

The *trusted-server* scheme depends on a trusted server for key agreement between nodes, e.g., Kerberos [18]. This type of scheme is not suitable for WSNs because there is usually no trusted infrastructure in sensor networks. The *self-enforcing* scheme depends on asymmetric cryptography, such as key agreement using public key certificates. However, limited computation and energy resources of sensor nodes often make it undesirable to use public key algorithms, such as Diffie-Hellman key agreement [19] or RSA [20], as pointed out in [10]. The third type of key agreement scheme is key *pre-distribution*, where key information is distributed among all sensor nodes prior to deployment. If we know which nodes are more likely to stay in the same neighborhood before deployment, keys can be decided *a priori*. However, because of the randomness of the deployment, knowing the set of neighbors deterministically might not be feasible.

B. Key Type

In order to support different communication patterns, the following types of keys are useful in sensor networks:

- *Node keys:* A node key is a key that is shared by a sensor node and the base station. It is used to protect unicast messages exchanged between the sensor node and the base station that do not need in-network processing.
- *Link keys:* A link key is a key shared by two neighboring nodes (i.e., two sensor nodes or a sensor node and the base station). Link keys provide protection for unicast messages exchanged between neighboring nodes. They can be used for encryption, message authentication, and integrity protection. They allow for in-network processing by hop-by-hop protection of data packets sent from the sensor nodes to the base station. They can also be used to set up other keys between neighboring nodes, such as cluster keys.
- *Cluster keys:* A cluster key is a key shared by a node and all of its neighbors. This key is used to encrypt (and decrypt) local broadcast messages. In addition, hop-by-hop encryption of a data packet with local broadcast keys makes passive participation possible, as it ensures that the neighbors of the transmitting nodes can learn the content of the packet.
- *Network key:* The network key is a key that is shared by all the nodes in the network. It is used to encrypt (and decrypt) global broadcast messages.

Note that cluster keys and the network key are broadcast keys, and they cannot be used for message authentication. The reason is that a node receiving a message with a message authentication code computed with a broadcast key, cannot identify who the originator of the message is; indeed, any node that possesses the broadcast key may have sent the message.

V. Energy Analysis of Public -Key Cryptosystem

In this section, we will try to estimate the performance ratio between WSN nodes for energy analysis of public key cryptosystem focusing on cryptographic calculations.

We setup our simulations as follows: The network is supposed to be connected. The objectives of our simulations are to compare the energy cost of public key algorithm and key management schemes based on public key cryptography. To perform our experiments, we deploy several hundred nodes (250 to 640) in random topology and run the key setup phase. The sensor node we use for our simulation is MICA2DOT based on the ATmega128L microcontroller from ATMEL.

Arvinderpal S. [21] provides detailed measurements for the MICA2DOT. They have measured power consumption for the MICA2DOT for the following cryptographic operations:

- Signature generation/verification and client/server key exchange operations(see Table III),
- Calculation of SHA-1 hash value (5.9 μWs),

TABLE III
Energy Cost of RSA and ECC (mW) [21]

Algorithm	Key Size	Key Exchange		Signature	
		Client	Server	Sign	Verify
RSA	1024	15.4	304	304	11.9
	2048	57.2	2302.7	2302.7	53.7
ECC	160	22.3	22.3	22.82	45.09
	224	60.4	60.4	61.54	121.98

However, in [21] the power consumption of active MICA2DOT is said to be 13.8 mW. We used the power consumption model presented in [21] to estimate the time needed by MICA2DOT. These data are used to calculate the time and power consumption for all other nodes.

The ATmega128 microcontroller current consumption is presented in Table IV for three different modes of operation, two different clock speeds and two supply voltages [22].

TABLE IV
The ATmega128 Microcontroller Current Consumption [22]

Mode			consumption
Active	4MHz	3.3V	6mA
		5V	9 mA
	8MHz	3.3V	10 mA
		5V	17 mA
Idle	4MHz	3.3V	2 mA
		5V	4 mA
	8MHz	3.3V	4 mA
		5V	8 mA
Power-Save	-	3.3V	9μA
		5V	13μA

If the microcontroller is running in active mode on 8 MHz and 3.3 V, then the energy consumed is:

*3.3 V * 10 mA=33 mW*

33 mW/ 8 MHz = 4.125 nW/clock cycle (nWs)

The times elapsed for signature generation, verification and key exchange for the client and server side, when the

active power consumption is equal to 13.8 mJ, are given in Table V.

TABLE V
TIME CONSUMPTION VERSUS ALGORITHM AND KEY SIZE [8]

Algorithm	Key Size (bit)	Key Exchange		Signature	
		Client (s)	Server (s)	Sign (s)	Verify (s)
RSA	1024	1.12	22.03	22.03	0.86
	2048	4.14	166.85	166.85	3.89
ECC	160	1.62	1.62	1.65	3.27
	224	4.38	4.38	4.46	8.84

Now we can calculate the power consumption for signature generation, verification and key exchange based on the results of Table IV and Table V. Therefore the active power consumption is equal to 33 mJ (see Table VI).

TABLE VI
THE ESTIMATED POWER CONSUMPTION (MWS)

Algorithm	Key Size	Key Exchange		Signature	
		Client	Server	Sign	Verify
RSA	1024	39.96	726.99	726.99	28.38
	2048	136.62	5506.05	5506.05	128.37
ECC	160	53.46	53.46	54.45	107.91
	224	144.54	144.54	147.18	291.72

VI. CONCLUSION

Our calculations show that RSA is not well suited for WSNs. Comparing ECC-160 and RSA-1024 indicates that the effort needed for RSA cryptography is rather too much. While the application of the even stronger ECC-224 still seems to be feasible, the time and power consumption for the equivalent RSA-2048 is far beyond the acceptable level.

In addition to key management and secure communication, public-key cryptography can be the enabling technology for numerous other WSN applications, including securely connecting pervasive devices to the Internet and distributing signed software patches.

REFERENCES

[1] D. Djenouri, L. Khelladi, "A Survey of Security Issues in Mobile Ad Hoc and Sensor Networks," IEEE Communication Surveys and Tutorials, vol. 7, no. 4, pp. 2–28, December 2005.
[2] G. Gaubatz, J-P. Kaps, B. Sunar, "Public Key Cryptography in Sensor Networks Revisited", 1st European Workshop on Security in Ad-Hoc and Sensor Networks (ESAS 2004), Lecture Notes in Computer Science, vol. 3313, Springer, Heidelberg, pp. 2-18, August, 2004.
[3] A. Perrig, R. Szewczyk, V. Wen, D. Culler, J. D. Tygar, "SPINS: Security Protocols for Sensor Networks," Wireless Networks, vol. 8, no. 5, pp. 521–534, September 2002.
[4] A. Baggio, "Wireless sensor networks in precision agriculture", in ACM Workshop on Real-World Wireless Sensor Networks (REALWSN 2005), Stockholm, Sweden, June 2005.
[5] A. Mainwaring, J. Polastre, R. Szewczyk, D. Culler, J. Anderson, "Wireless Sensor Networks for Habitat Monitoring", in First ACM Workshop on Wireless Sensor Networks and Applications, Atlanta, GA, USA, September 2002.
[6] G. Fuchs, S. Truchat, F. Dressler, "Distributed Software Management in Sensor Networks using Profiling Techniques", in 1st IEEE/ACM International Conference on Communication System Software and Middleware (IEEE COMSWARE 2006): 1st International Workshop on Software for Sensor Networks (SensorWare 2006), New Dehli, India, January 2006.
[7] W. Zhang, G. Cao, "Group Rekeying for Filtering False Data in Sensor Networks: A Predistribution and Local Collaboration-Based Approach", in 24th IEEE Annual Joint Conference of the IEEE Computer and Communications Societies (IEEE INFOCOM 2005), pp. 503–514, March 2005.
[8] K. Piotrowski, P. Langendoerfer, S. Peter, "How Public Key Cryptography Influences Wireless Sensor Node Lifetime", Proceedings of the fourth ACM workshop on Security of ad hoc and sensor networks, USA, pp.169 – 176, 2006
[9] Y. W. Law, "Key Management and Link-Layer Security of WSN", Ph.D. Thesis, University of Twente, Netherland, 2005.
[10] A. Perrig, R. Szewczyk, V. Wen, D. Culler, J. D..Tygar. "SPINS: Security Protocols for Sensor Networks," in Proceedings of the 7th Annual ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom), Rome, Italy, pp. 189–199, July 2001..
[11] S. Zhu, S. Setia, S. Jajodia. "LEAP: Efficient Security Mechanisms for Large-Scale Distributed Sensor Networks", In the Proceedings of the 10th ACM conference on Computer and communications security, 2003.
[12] J. P. Walters, Zh. Liang, W. Shi, V. Chaudhary, "Security in Distributed, Grid, and Pervasive Computing", Chapter 17, CRC Press, 2006.
[13] R. B. Ghazali, "Security in WSN in Enhance AODV Routing", Masters thesis, Faculty of Electrical Engineering, University Technology Malaysia, 2006.
[14] N. Koblitz, "Elliptic curve cryptosystems", Mathematics of Computation, Vol. 48, 1987.
[15] V.S. Miller, "Use of Elliptic Curves in Cryptography", Advances in Cryptology CRYPTO 85, 1986.
[16] A. J. Menezes, P. C. van Oorschot, S. A. Vanstone, "Handbook of Applied Cryptography", CRC Press, 1996.
[17] W. Du, J. Deng, Y. S. Han, Shigang Chen, P.K. Varshney, "A Key Management Scheme for Wireless Sensor Networks Using Deployment Knowledge", IEEE INFOCOM 2004.
[18] B. C. Neuman, T. Tso, "Kerberos: An authentication service for computer networks", IEEE Communications, vol. 32, no. 9, pp. 33–38, September 1994.
[19] W. Diffie, M. E. Hellman, "New directions in cryptography", IEEE Transactions on Information Theory, vol. 22, pp. 644–654, November 1976.
[20] R. L. Rivest, A. Shamir, and L. M. Adleman, "A method for obtaining digital signatures and public-key cryptosystems", Communications of the ACM, vol. 21, no. 2, pp. 120–126, 1978.
[21] A. S. Wander, N. Gura, H. Eberle, V. Gupta, Sh. Ch. Shantz, "Energy analysis of public-key cryptography for wireless sensor networks", In PERCOM '05: Proceedings of the Third IEEE International Conference on Pervasive Computing and Communications, pp. 324–328, Washington, DC, USA, 2005. IEEE Computer Society.
[22] ATmega128(L) Data Sheet (2006). Atmel Corporation, Available at: http://www.atmel.com/dyn/resources/prod_documents/doc2467.pdf