Available online at www.sciencedirect.com

SciVerse ScienceDirect

Procedia Technology 6 (2012) 978 – 987

Procedia Technology

2nd International Conference on Communication, Computing & Security [ICCCS-2012]

Security Frameworks for Wireless Sensor Networks-Review

Gaurav Sharma[a]* Suman Bala[a], Anil K. Verma[a]

[a]*Thapar University, Patiala-147004, India*

Abstract

Due to significant advances in miniaturization, low power circuit design but reasonably efficient to carry the sensitive information through wireless communication, wireless sensor network (WSN) have attracted attention a lot in recent years. WSN's are being used in many applications like health monitoring, military purposes, and home automation. Since WSN suffer from many constraints including lower processing power, low battery life, small memory and wireless communication channel, security becomes the main concern to deal with such kind of networks. Due to these well accepted limitations, WSN is not able to deal with traditional cryptographic algorithms. This paper gives an overview of cryptographic frameworks designed so far and also a comparison of existing schemes is tabled.

Keywords: Wireless Sensor Network, Security; Cryptography; Survey; Frameworks;

1. Introduction

The field of sensor network is well known due to its popularity in research community. It is a collection of thousands of self-organized sensor nodes capable of wireless communication. Since the nodes are not so wealthy in terms of resources, therefore complex algorithms cannot be played over it. Security is the main pre-concern to socialize this network for common usage. For making the WSN secure, cryptography plays an important role. There are many algorithms proposed so far: symmetric, asymmetric and hybrid. But complex algorithms, which had been proposed for MANETs, are not successful over WSN.

To design the network cryptographically (completely) secure, security must be integrated into every node

* Corresponding author. Tel.: +91-9780895083.
E-mail address: sharmagaurav@ieee.org

doi:10.1016/j.protcy.2012.10.119

of the network. So security should be implemented at every point of the network. Cryptography is a standard method to provide security in a network. But here in WSN, cryptographic algorithms should be designed such that it is robust in nature but does not use more memory, more power, and more energy so as the lifetime of the network can be increased. So the designer's job is here more challenging, but here also we can say, the security is also dependent upon the nature of the application and algorithm might be specific to the application.

1.1. Security Requirements

Confidentiality - Confidentiality ensures the concealment of the message from an attacker so that any message communicated via the sensor network remains confidential. In a WSN, the issue of confidentiality should address the following requirements: (i) a sensor node should not allow its readings to be accessed by its neighbors unless they are authorized to do so, (ii) key distribution mechanism should be extremely robust, (iii) public information such as sensor identities, and public keys of the nodes should also be encrypted in certain cases to protect against traffic analysis attacks.

Authentication - Authentication ensures the reliability of the message by identifying its origin. By authenticating other nodes, cluster heads, and base stations before granting a limited resource, or revealing information. In a WSN, the issue of authentication should address the following requirements: (i) communicating node is the one that it claims to be, (ii) receiver node should verify that the received packets have undeniably come from the actual sender node.

Integrity - Integrity ensures the reliability of the data and refers to the ability to confirm that a message has not been tampered with, altered or changed while on the network. In a WSN, the issue of integrity should address the following requirements: (i) only the nodes in the network should have access to the keys and only an assigned base station should have the privilege to change the keys. This would effectively thwart unauthorized nodes from obtaining knowledge about the keys used and preclude updates from external sources. (ii) It protects against an active, intelligent attacker who might attempt to disguise his attack as noise.

Availability - Availability ensures the services of resources offered by the network, or by a single sensor node must be available whenever required. In a WSN, the issue of availability should address the following requirements: (i) the security mechanisms should be available all the time; a single point of failure should be avoided, (ii) the mechanism is used as a central access control system to ensure successful delivery of every message to its recipient node.

1.2. Significance of Cryptography in Wireless Sensor Networks

Since the popularity of WSN increasing for a wide variety of applications such as climate change, environmental monitoring, traffic monitoring and home automation. So to keep the WSN secure is a challenging task. Cryptography is one way to provide security. It can be provided through by symmetric key techniques, asymmetric key techniques and hash function. Since WSN are very constrained in terms of computing, communication and battery power, it requires a light weight cryptographic algorithm. Due to constraints of sensor nodes, the selection of cryptographic technique is vital in WSN. Cryptography in WSN can be explained in the following three aspects: symmetric, asymmetric and hash function.

2. Types of Cryptographic Techniques

It is important to select the most appropriate cryptographic method because all the security requirements are ensured by cryptography. Cryptographic methods used in WSNs should meet the constraints of sensor nodes and be evaluated by code size, data size, processing time, and power consumption. However, sensor nodes are limited in their computational and memory capabilities, so the traditional cryptographic techniques cannot be simply transferred to WSNs. Consequently, to meet the above mentioned security requirements, either the existing techniques have to be adapted or novel techniques have to be developed. Based on the existing cryptographic techniques, we can classify them into three classes: symmetric cryptographic techniques, asymmetric cryptographic techniques and hybrid cryptographic techniques. Asymmetric cryptographic techniques can further be classified into three classes: RSA based techniques, ECC based techniques and pairing based techniques. In this section we discuss the various types of cryptographic techniques evaluated for wireless sensor networks until now.

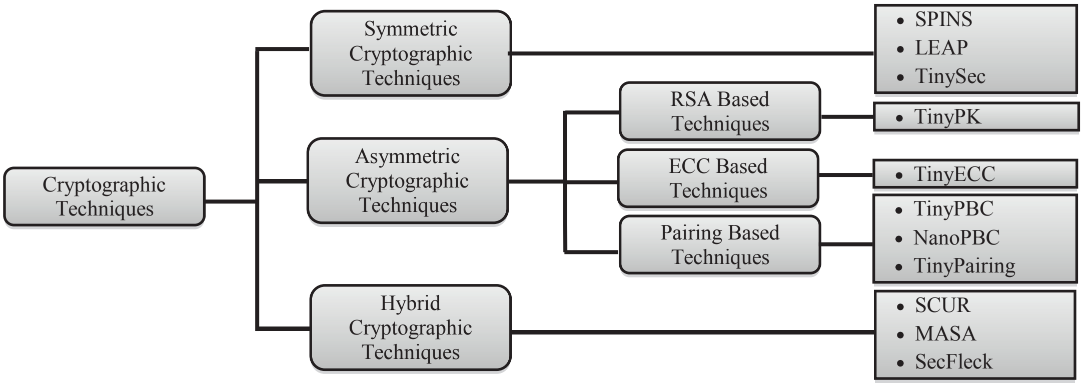

Fig. 1. Cryptographic Techniques

2.1. Symmetric Cryptographic Techniques

In symmetric cryptographic techniques, a single shared key is used between the two communicating nodes both for encryption and decryption. This key has to be kept secret in the network, which can be quite hard in the exposed environment where WSNs are used. Most security schemes for WSN use only symmetric cryptography, due to its ease of implementation on limited hardware and small energy demands, especially if the implementation is done in hardware to minimize performance loss. Two types of symmetric ciphers are used: block ciphers that work on blocks of a specific length and stream ciphers that work bitwise on the data. A steam cipher can be seen as a block cipher with a block length of 1 bit.

Ganeshan et al., 2003 had presented a survey investigating the computational requirements for a number of popular cryptographic algorithms and presented the impact of a variety of encryption techniques for embedded architectures instead of general-purpose processors. Five popular encryption schemes RC4, RC5, IDEA, SHA-1, and MD5, were evaluated on six different microprocessors ranging in word size from 8-bit (Atmel AVR) to 16-bit (Mitsubishi M16C) to 32-bit widths (StrongARM, XScale). RC4 outperforms RC5 on encryption in low-end processors, compared to the choice of RC5 for current sensor networks. Law et al., 2006 evaluate a survey of block ciphers for WSNs. In this paper, they evaluated two symmetric key algorithms: RC5 and TEA. They further evaluated six block ciphers: RC5, RC6, Rijndael, MISTY1, KASUMI, and Camellia on IAR Systems' MSP430F149. Passing et al., 2006 verified with an experimental setup the runtime behaviour of cryptographic algorithms for WSNs including MD5, SHA-1 and AES. For hashing of arrays of different sizes MD5 outperforms SHA-1. The required time is linearly dependent to the amount of data. The overall results show that the examined cryptographic algorithms need high execution times of a single AES operation on a 1kByte array lasted 1.67s. Fournel et al., 2007 evaluate a survey of stream ciphers for WSNs. They evaluated stream cipher algorithms: DRAGON, HC-256, HC-128, LEX, Phelix, Py and Pypy, Salsa20, SOSEMANUK are all dedicated to software uses and were originally submitted to the European Project Ecrypt in the eStream call (Phase 2). Choi and Song, 2006 investigated the feasibility of various cryptographic algorithms, AES, Blowfish, DES, IDEA, MD5, RC4, RC5, SEED, SHA-1 and SHA-256, for their use in WSN utilising MICAz-type motes running TinyOS. The usage of resources including memory, computation time and power for each cryptographic algorithm were experimentally analysed. As a result, RC4 and MD5 turned out as the most suitable algorithms for MICAz-type motes. Healy et al., 2008 had a look on the benefits of using hardware encryption for symmetric encryption to reduce encrypting costs. Thus, the authors investigated the use of the AES encryption module that is available on the Chipcon CC2420 transceiver chip as used for MICAz and TmoteSKY nodes. Three symmetric cryptographic algorithms were examined: AES, RC5 and Skipjack, AES outperforms better in terms of less memory use, increased execution time and less power consumption.

2.2. Asymmetric Cryptographic Techniques

In asymmetric cryptography, a private key can be used to decrypt and sign data while a public key can be used to encrypt and verify data. The private key needs to be kept confidential while the public key can be published freely. Asymmetric cryptography is also known as Public key cryptography. Public key cryptography tends to be resource intensive, as most systems are based on large integer arithmetic. For a number of years many researchers discarded public key cryptography as infeasible in the limited hardware used in WSN. The code size, data size, processing time, and power consumption make it undesirable for public key algorithm techniques, such as the Diffie-Hellman key agreement protocol or RSA signatures, to be employed in WSNs. There are various public key algorithms include Rabin's Scheme, Ntru-Encrypt, RSA, Elliptic Curve Cryptography (ECC), Pairing Based Cryptography (PBC) and Identity Based Encryption.

Malan et al., 2004 presented the first known implementation of elliptic curve cryptography over F_{2p} for sensor networks based on the 8-bit, 7.3828-MHz MICA2 mote. Through analysis of their implementation for TinyOS of multiplication of points on elliptic curves, they argued that public-key infrastructure is viable for TinySec keys' distribution, even on MICA2. They have implemented EccM 1.0 which is a TinyOS module based on code by Michael Rosing, rather it was a failure. Further, they implemented EccM 2.0 based on the design of Dragongate Technologies Limited's Java-based jBorZoi 0.9 which was a success. EccM 2.0 employed much less memory than EccM 1.0, its running time bests that for Diffie-Hellman based on DLP, using keys an order of magnitude smaller in size but no less secure. Gaubatz et al., 2004 challenged the basic assumptions about public key cryptography in sensor networks which are based on a traditional software based approach. They also proposed a custom hardware assisted public key approach which is based on optimised algorithms and associated parameters as well as low-power design. They compared two architectures that implement two different types of public key crypto-systems. The first one, Rabin's Scheme, is a specialization of the well-known RSA algorithm where the exponent is fixed to the value 2. The second algorithm, NtruEncrypt, is a public key cryptosystem where security is based on the hardness of the Shortest Vector Problem (SVP) in a very high dimension lattice. Gaubatz et al., 2005 improved the energy consumption of public key cryptography. They proposed a custom hardware assisted approach using a special purpose ultra-low power hardware implementations of public key algorithms to make public key cryptography feasible for WSNs. They conclude that the use of public key cryptography results in a reduced protocol overhead, less packet transmissions and thus, power savings. Besides, an in-depth comparison of three unalike public key cryptography implementations for WSNs is provided including Rabin's Scheme, NtruEncrypt or and ECDSA/ECMV. Wander et al., 2005 quantified the energy cost of authentication and key exchange based on public-key cryptography; RSA and ECC on an 8-bit microcontroller platform; Atmel ATmega128 processor. A comparison has been presented on two public-key algorithms, RSA and Elliptic Curve Cryptography (ECC), and considers mutual authentication and key exchange between two untrusted parties such as two nodes in a wireless sensor network. The ECC-based signature is generated and verified with the Elliptic Curve Digital Signature Algorithm (ECDSA). The results have shown that ECDSA signatures are significantly cheaper than RSA signatures. The experiments were conducted on the Berkeley/Crossbow motes platform, specifically on the Mica2dots. The implementation of RSA and ECC cryptography on Mica2 nodes further proved that a public key-based protocol is viable for WSNs.

Arazi et al., 2005 described the efficiency of public-key cryptography for WSNs and the corresponding issues that need to be considered. Particularly, ECC is highlighted as suitable technique for WSN which provides a good trade-off between key size and security. Lopez, 2006 focused on the security issues by analysing the use of symmetric cryptography in contrast with public-key cryptography. The author also discussed the important role of elliptic curve cryptography in this field. Wander et al., 2005 quantified the energy cost of authentication and key exchange based on public key cryptography on an 8-bit Atmel ATmega128L low-power microcontroller. They compared the two public key algorithms RSA and ECC, considering mutual authentication between two parties. The result shows that even software-based public key cryptography is feasible for an 8-bit microcontroller platform. However, ECC shows significant better results as compared to RSA in terms of reduced computation time, amount of data that needs to be stored and transmitted. Consequently, ECC requires less energy than RSA. Piotrowski and Peter, 2006 estimated the power consumption of the most common RSA and ECC operations, such as signature generation and verification, as well as the involved transmissions on common sensor platforms such as MICA2DOT, MICA2, MICAz and TelosB. The result shows that public key cryptography does not influence the sensors lifetime significantly, so that strong cryptography can be seen as feasible for WSNs. Batina et al., 2006 proposed a low-cost public key cryptography scheme for sensor networks providing service such as key distribution and authentication. They proposed a custom hardware assisted approach to implement Elliptic Curve Cryptography (ECC) in order to obtain stronger cryptography as well as to minimize the power. The low-power ECC processor contains a modular arithmetic logical unit (MALU) for ECC field arithmetic.

Szczechowiak et al., 2008 presented results on implementing ECC, as well as the related emerging field of Pairing-Based Cryptography (PBC), on two of the most popular sensor nodes MICA2 and Tmote Sky. They showed that ECC over prime field is not always the best option as pairings over $GF(2^m)$ seem to be more efficient on this type of architecture. They argued that fast pairing computation enables Identity Based Encryption and thus opens new ways for achieving security in sensor networks which was also argued by Oliveira et al., 2007. Yeh et al., 2011 proposed ECC-based user authentication protocol that resolves some weaknesses.

2.3. Hybrid Cryptographic Techniques

Symmetric and asymmetric cryptography can be applied in combination to join the advantages of both approaches. Pugliese and Santucci, 2008 discussed a novel hybrid cryptographic scheme for the generation of pairwise network topology authenticated keys (TAK) in WSNs, which is based on vector algebra in $GF_{(q)}$. Symmetric is used for ciphering and authentication, while asymmetric is used for key generation.

3. Cryptographic Frameworks for Wireless Sensor Networks

In this section, we discussed frameworks which are specifically designed and implemented to provide security to wireless sensor networks. We classify the existing frameworks according to the nature of the key material i.e. the shared key is private or public. We further classify asymmetric cryptographic frameworks into three classes as RSA based cryptographic frameworks, ECC based cryptographic frameworks and pairing based cryptographic frameworks.

3.1. Symmetric Cryptographic Frameworks

In this subsection we discuss cryptographic frameworks which are based on single shared key both for encryption and decryption. Such frameworks are: SPINS, Localized Encryption and Authentication Protocol (LEAP) and TinySec.

- *SPINS* - Perrig et al., 2001 proposed a security building block, which is optimized for resource-constrained environments and wireless communication. It based on two secure building blocks: SNEP (Secure Network Encryption Protocol) and µTESLA (the "micro" version of the Timed, Efficient, Streaming, Loss-tolerant Authentication Protocol). SNEP provides the data confidentiality, two-party data authentication, and data freshness with less communication cost. While µTESLA provides authenticated broadcast for severely resource-constrained environments. The scheme is resilient to node capture and possible to revoke key. But it is not scalable and the base station becomes the target of attacks. It does not provide a solution for denial of service (DoS) attacks when the malicious node keeps sending the request to negotiate a session key because one adversary can easily trigger a REPLAY attack and exhaust the energy in the sensor nodes.
- *Localized Encryption and Authentication Protocol (LEAP)* - Zhu et al., 2003 proposed a protocol that was designed to support in-network processing. The design of the protocol was based on the principle that different types of messages exchanged between sensor nodes have different security requirements; a single keying mechanism is not suitable for meeting these different security requirements. Evaluation of the code was done on Mica2 motes. It doesn't nclude the issues of packet loss and feedback implosion. LEAP had analysed under various attack models and it comes out to be very effective in defending against many sophisticated attacks such as HELLO Flood attack, Sybil attack, and Wormhole attack. It is based on some assumptions that a pre-distributed key shared among all nodes will not be disclosed during the *t* initial time units of the network operation. And once this key is erased, it cannot be recovered from memory.
- *TinySec* - Karlof et al., 2004 introduced a lightweight, generic security package that developers can easily integrate into sensor network applications. It is the first fully-implemented protocol for link-layer cryptography in sensor networks. The implementation of TinySec is incorporated into the official TinyOS release. It includes some of the trade-offs between performance, transparency, and cryptographic security and a design is based on the needs of applications in the sensor network space. Bandwidth, latency, and energy costs of TinySec are low for sensor network applications. TinySec is easily extensible and has been incorporated into higher level protocols.

3.2. Asymmetric Cryptographic Frameworks

In this subsection, we discuss cryptographic frameworks, which are based on two shared keys, private key for encryption and public key for decryption. Asymmetric cryptographic frameworks are further classified as RSA based, ECC based and Pairing based cryptographic frameworks.

3.2.1 *RSA Based cryptographic frameworks*

In this subsection of asymmetric cryptographic frameworks, we discuss cryptographic frameworks, which uses RSA (Rivest-Shamir-Adleman) algorithm for the security measures. RSA is computationally intensive and usually execute thousands or even millions of multiplication instructions to perform a single-security operation. The number of clock cycles required to perform a multiplication instruction primarily determines a microprocessor's public key algorithm efficiency. Carman et al., 2000 found that it usually takes a microprocessor thousands of nano-joules to do a simple multiplication function with a 128-bit result.

- *TinyPK* - Watro et al., 2004 have described the design and implementation of public-key-based protocols; that allow authentication and key agreement between a sensor network and a third party as well as between two sensor networks. TinyPK incorporating the use of TinySec provides the functionality needed for a mote and a third-party to mutually authenticate to each other and to communicate securely. TinyPK is based on the well-known RSA cryptosystem, using e=3 as the public exponent.

3.2.2 *ECC Based cryptographic frameworks*

In this subsection of asymmetric cryptographic frameworks, we discuss cryptographic frameworks, which uses ECC (Elliptical Curve Cryptography) algorithm for the security measures. The security of ECC is based on the elliptic curve discrete logarithm problem, which the cryptographic community regards as much more difficult than the integer factorization and discrete logarithm problems that underlie the conventional Rivest-Shamir-Adelman (RSA) and Diffie–Hellman public-key algorithms. ECC has two main advantages: (1) ECC public keys are smaller for the same level of security as RSA or Diffic–Hellman-based solutions, thus reducing the number of bits that need to be exchanged; and (2) ECC public-key operations require fewer computations than conventional public-key methods. The benefit of smaller key is that they need less storage, less bandwidth and, therefore, less energy, thereby reducing processing and communication overhead, which is ideal for energy-constrained sensor nodes.

- *TinyECC* - Liu and Ning, 2008 presented the design, implementation, and evaluation of TinyECC, a configurable library for ECC operations in wireless sensor networks. The primary objective of TinyECC is to provide a ready-to-use, publicly available software package for ECC-based PKC operations that can be flexibly configured and integrated into sensor network applications. TinyECC provides a number of optimization switches, which can turn specific optimizations on or off based on developers' needs. Different combinations of the optimizations have different execution time and resource consumptions, giving developers great flexibility in integrating TinyECC into sensor network applications.

3.2.3 *Pairing Based cryptographic frameworks*

In this subsection of asymmetric cryptographic frameworks, we discuss a cryptographic framework which uses pairing based cryptography for the security measures. Cryptography using Pairings (PBC) is an emerging field related to ECC, which has been attracting the interest of international cryptography community, since it enables the design of original cryptographic schemes and makes well-known cryptographic protocols more efficient.

- *TinyPBC* - Oliveira et al., 2008 proposed TinyPBC, which is based on Multi-precision Integer and Rational Arithmetic C/C++ Library (MIRACL) which is a publicly available and open source library written in C. the authors demonstrates how sensor nodes can exchange keys in an authenticated and non-interactive way. They also present the fastest pairing computation on an 8-bit platform, which shows the best figures for binary field multiplication on an 8-bit platform. The result showed TinyPBC takes only 5.45s to compute pairings on ATmega128L. TinyPBC requires less than half the amount of time of the NanoECC result, which takes 10.96s to compute pairings. The source code is available at www.lca.ic.unicamp.br/~loliveira#tinypbc.
- *NanoPBC* - Aranha et al., 2009 proposed a cryptographic library for resource-constrained devices. The authors implemented all big number, finite field, and elliptic curve arithmetic from scratch therefore it would allow to extract the most from the platform. Besides, in order to maximize speed, all time critical routines were implemented using Assembly. Another key factor for achieving the performance was the implementation of the L´opez-Dahab binary finite field multiplication. It was optimized for minimizing memory access operations, which are expensive on our target platform. The authors provide an in-depth description of how to implement pairings efficiently in resource-constrained devices and presented the fastest figures for pairing computation on an 8-bit platform.
- *TinyPairing* - Xiong et al., 2010 proposed an efficient and lightweight pairing-based cryptographic library for sensors. It provides a better way to compute quickly as it consumes low memory for both the cases RAM and ROM by deliberately choosing some super-singular elliptic curve as the pairing group and some specific finite field, which defines the elliptic-curve pairing group. The source code is available at http://www.cs.cityu.edu.hk/~ecc/TinyPairing. The authors had evaluated the average running time. The RAM and ROM usage by each pairing-based scheme were obtained using the TinyOS tool-chain.

Table 1. Security Frameworks for Wireless Sensor Networks.

Framework	*Encryption*	*Cipher*	*Freshness (CTR)*	*Key Agreement*	*Code Requirement*	*Authentication*	*Cost (time/energy)*	*Support*
SPIN	CTR mode	RC5 (Block)	Yes	Master Key & Delayed Disclosure	2674B	CBC-MAC	7.24 ms	SmartDust
LEAP	RC5	RC5 (Block)	No	Pre-deployed (Master Varaible)	ROM: 17.9KB RAM: no. of neighbours	CBC-MAC	Variable (No. of neighbours)	Mica2
TinySec	CBC mode (Optional)	Cipher independent	No	Any	RAM: 728B program space: 7146B	CBC-MAC	RC5(C): 0.90ms Skipjack(C): 0.38ms RC5(C, assembly): 0.26ms	Mica, Mica2, & Mica2Dot
TinyPK	RSA	-	No	PK-RSA	13387B (512 bit key)	CA-signed Diffie-Hellman public value	3.8 s	Mica1, Mica2
TinyECC	ECIES	-	No	ECDH	20818B (micaz)	ECDSA	20266.47ms / 486.4 mJ (micaz)	Mica2/MicaZ, TelosB/Tmote Sky, BSNV3, & Imote2
TinyPBC	PBC	-	No	ID-NIKDS	Stack: 2,867B RAM: 368B ROM: 47,948B	ID-NIKDS	5.45s (pairing computation)	Mica2 & MicaZ
NanoPBC	-	-	-	-	-	-	-	MicaZ
TinyPairing	PBC	-	No	-	RAM: 392B ROM: 21,742B	-	21.95 s	MICA family, Telos, eyesIFX, Intel's Imote,
SCUR	Rabbit	Rabbit (Stream) (128bit)	Yes	Pre-Deployed key	-	-	-	-
MASA	-	-	No	-	-	-	-	-
SecFleck	RSA	RSA (Block) (2048 bit) & XTEA (128 bit)	No	-	RAM: 52B Program space: 1,082B	RSA	RSA (s/w): 219,730 μs / 7,030.0 μ J RSA (h/w): 27 μ s / 5.4 μ J XTEA (s/w): 18 μ s / 0.6 μ J	Fleck sensor node

3.3. Hybrid Cryptographic Frameworks

In this subsection, we discuss cryptographic frameworks, which are based on the combination of two approaches; symmetric cryptography and asymmetric cryptography. In 2011, Jailin, 2011 published a novel scheme known as Dynamically Secured Authenticated and Aggregation scheme (DSAA), which is a combination of public and symmetric key cryptography, leads to increase in security, computational speed as well as reduces memory usage.

- *SCUR* - Tahir et al., 2008 proposed a Lightweight Encryption Mechanism based on the Rabbit stream cipher for providing confidentiality in Wireless Sensor Networks. It fulfils both requirements of security as well as energy efficiency. The objective of the SCUR is to minimize cost-effect of the following while maintaining required levels of security: (1) Communication overhead, in case of communicating the encrypted packet. (2) Computation overhead, in securing the network, in order to save sensor's lifetime. (3) Utilized key space.

- *MASA* - Alzaid et al., 2008 proposed a security system known as MASA (Mixture of Asymmetric and Symmetric Approaches) to provide end-to-end data security for wireless sensor networks. It is based on the concept of virtual geographic grid wherein the entire terrain is broken down into smaller regions called cells. Each sensor carries two types of keys, asymmetric and symmetric. MASA uses the private key to sign a hashed event notification to provide confidentiality, authenticity, and data integrity. The symmetric key is used to authenticate the event notification within its cell.
- *SecFleck* - Hu et al., 2009 described the design and implementation of a public-key platform. It is based on a commodity Trusted Platform Module (TPM) chip that extends the capability of a standard node. SecFleck chooses XTEA symmetric key cryptography because of its small RAM footprint, which makes it a good candidate for tiny sensor devices that typically have less than 10 KB RAM. XTEA can be used in an output feedback mode to encrypt or decrypt variable length strings. The author's had discussed the performance of the secFleck platform in terms of computation time, energy consumption, and financial cost.

4. Conclusion

The wireless sensor networks continue to grow and become widely used in many applications. So, the need for security becomes vital. However, the wireless sensor network suffers from many constraints such as limited energy, processing capability, and storage capacity, etc. Consequently, many innovative security protocols and techniques have been developed to meet this challenge. There are many ways to provide security, one is cryptography. Selecting the appropriate cryptography method for sensor nodes is fundamental to provide security services in WSNs. In this paper, we present an analysis of cryptographic frameworks designed so far. Also a comparison of proposed schemes is presented. Table 1 presents the various frameworks for different parameters like encryption, ciphering, freshness, key agreement, code requirement, authentication, cost and which mote supports this framework. This comparison chart might be helpful for research community and might provide a future direction to them also.

References

Alzaid, H., Alfaraj, M., 2008. "MASA: End-to-End Data Security in Sensor Networks Using a Mix of Asymmetric and Symmetric Approaches," 2nd IEEE International Conference on New Technologies, Mobility and Security.

Aranha, D., Lopez, J., Oliveira, L., Dahab, R., 2009. "NanoPBC: Implementing Cryptographic Pairings on an 8-bit Platform," Conference on Hyperelliptic curves, discrete Logarithms, Encryption, etc. (CHiLE 2009), Frutillar, Chile.

Arazi, B., Elhanany, L., Arazi, O., Qi, H., 2005. Revisiting public-key cryptography for wireless sensor networks. IEEE Computer, 38(11): 103–105.

Batina, L., Mentens, N., Sakiyama, K., Preneel, B., Verbauwhede, I., 2006. Low-cost elliptic curve cryptography for wireless sensor networks. Lecture Notes in Computer Science, 4357: 6–17.

Carman, D., Kruus, P., Matt, B., 2000. Constraints and approaches for distributed sensor network security. NAI Labs, TR.

Choi, K., Song, J., 2006. "Investigation of feasible cryptographic algorithms for wireless sensor network," 8th ICACT-2006, 2.

Fournel, N., Minier, M., Ubeda, S., 2007. Survey and Benchmark of Stream Ciphers for Wireless Sensor Networks. LNCS, 4462.

Ganesan, P., Venugopalan, R., Peddabachagari, P., Dean, A., Mueller, F., Sichitiu, M., 2003. "Analyzing and modeling encryption overhead for sensor network nodes," 2nd ACM Wireless Sensor Networks and Applications, New York, ACM Press.

Gaubatz, G., Kaps, J., Sunar, B., 2004. "Public key cryptography in sensor networks-Revisited," 1st ESAS '04.

Gaubatz, G., Kaps, J., Ozturk, E., Sunar, B, 2005. "State of the art in ultralow power public key cryptography for wireless sensor networks," 3rd IEEE International Conference on Pervasive Computing and Communications Workshops, Washington, DC, USA.

Healy, M., Newe, T., Lewis, E., 2008. Analysis of Hardware Encryption versus Software Encryption on Wireless Sensor Network Motes. Smart Sensors and Sensing Technology, 2008.

Hu, W., Corke, P., Shih, W., Overs, L., 2009. "SecFleck: A Public Key Technology Platform For Wireless Sensor Networks," 6th European Conference on Wireless Sensor Networks, Cork, Ireland.

Jailin, S., 2011. "Performance analysis of hybrid cryptography for secured data aggregation in wireless sensor networks," IEEE International Conference on Recent Trends in Information Technology, Chennai, 307-312.

Karlof, C., Sastry, N., Wagner, D., 2004. "TinySec: A Link Layer Security Architecture for Wireless Sensor Networks," 2nd ACM Conference on Embedded Networked Sensor Systems (SenSys 2004), Baltimore, MD.

Law, Y., Doumen, J., Hartel, P.,2006. Survey and benchmark of block ciphers for wireless sensor networks. ACM TOSN, 2(1).

Liu, A., Ning, P., 2008. "TinyECC: A configurable library for elliptic curve cryptography in wireless sensor networks," IPSN 2008,Washington, DC, USA, IEEE Computer Society.

Lopez, J., 2006. Unleashing public-key cryptography in wireless sensor networks. J of Computer Security, 14(5): 469–482.

Malan, D., Welsh, M., Smith, M., 2004. "A public-key infrastructure for key distribution in TinyOS based on elliptic curve cryptography," 1st IEEE Int. Conf. on Sensor and Ad Hoc Communications and Networks, Santa Clara, California.

Oliveira, L., Aranha, D., Morais, E., Daguano, F., Lopez, J., Dahab, R., 2007. "TinyTate: Computing the tate pairing in resource-constrained sensor nodes," 6th IEEE International Symposium on Network Computing and Applications.

Oliveira, L., Scott, M., Lopez, J, Dahab, R., 2008. "TinyPBC: Pairings for authenticated identity-based non-interactive key distribution in sensor networks," 5th International Conference on Networked Sensing Systems.

Passing, M., Dressler, F., 2006. "Experimental performance evaluation of cryptographic algorithms on sensor nodes," IEEE International Conference on Mobile Adhoc and Sensor Systems.

Perrig, A., Szewczyk, R., Wen, V., Culler, D., Tygar, J., 2001. "Spins: Security Protocols for Sensor Networks," 7th annual ACM/IEEE international conference on mobile computing and networking.

Piotrowski, K., Peter, S., 2006. "How public key cryptography influences wireless sensor node lifetime," 4th ACM workshop on Security of ad hoc and sensor networks. ACM New York, NY, USA.

Pugliese, M., Santucci, F., 2008. "Pair-wise network topology authenticated hybrid cryptographic keys for Wireless Sensor Networks using vector algebra," 5th IEEE International Conference on Mobile Ad Hoc and Sensor Systems (MASS 2008).

Sakai, R., Ohgishi, K., Kasahara, M., 2000. Cryptosystems based on pairing. Symposium on Cryptography and Info. Security.

Szczechowiak, P., Oliviera, L., Scott, M., Collier, M., Dahab, R., 2008. "NanoECC: Testing the limits of Elliptic Curve Cryptography in Sensor Networks," EWSN 2008, 4913: 305–320, LNCS, Springer-Verlag.

Tahir, R., Javed, M., Ahmad, A., Iqbal, R., 2008. "SCUR: Secure Communications in Wireless Sensor Networks Using Rabbit," World Congress on Engineering 2008, I, London, U.K.

Wander, A., Gura, N., Eberle, H., Gupta, V., Shantz, S., 2005. "Energy analysis of public-key cryptography for wireless sensor networks," 3rd IEEE International Conference on Pervasive Computing and Communication.

Watro, R., Kong, D., Cuti, S., Gardiner, C., Lynn, C., Kruus, P., 2004. "TinyPK: securing sensor networks with public key technology," 2nd ACM Workshop on Security of ad hoc and Sensor Networks (SASN'04).

Xiong, X., Wong, D., Deng, X., 2010. "TinyPairing: A Fast and Lightweight Pairing-based Cryptographic Library for Wireless Sensor Networks," WCNC 2010, IEEE Communications Society.

Yeh, H., Chen, T., Liu, P., Kim, T., Wei, H., 2011. A Secured Authentication Protocol for Wireless Sensor Networks Using Elliptic Curves Cryptography. Sensors, 4767-4779.

Zhu, S., Setia, S., Jajodia, S., 2003. "LEAP: Effcient Security Mechanisms For Large-Scale Distributed Sensor Networks," ACM Conference on Computing and Communication Security, CCS'2003.