Cryptography Methods for Software-Defined Wireless Sensor Networks

Sean W. Pritchard*, Gerhard P. Hancke* and Adnan M. Abu-Mahfouz*†
*Department of Electrical, Electronic and Computer Engineering
University of Pretoria Pretoria, South Africa
†Meraka Institute
Council for Scientific and Industrial Research (CSIR)
Pretoria, South Africa
Email: spritchard001@gmail.com, gerhard.hancke@up.ac.za, a.abumahfouz@ieee.org

***Abstract*—The Software-Defined Wireless Sensor Networking (SDWSN) paradigm aims to solve inherent issues present in Wireless Sensor Networks (WSN), such as resource constraints, by adopting a Software-Defined Networking (SDN) approach to the management of these WSNs. The security aspect of SDWSN has received little attention due to a focus on the architecture. As this paradigm is a combination of both WSN and SDN, some solutions from both paradigms can be adapted to consider SDWSN. One of the main problems with implementing security within WSN, lies within its inherent issues, such as resource constraints. However, due to the centralization of control brought about by the SDN paradigm, most of these issues are alleviated, leaving room for WSN security implementations. In order to investigate the use of WSN cryptography within SDWSN, cryptography methods have been implemented within a SDWSN network in order to verify whether the SDWSN paradigm does allow for resource intense WSN security implementations.**

***Index Terms*—IoT; WSN; security; security threats; SDN; SDWSN**

I. Introduction

Software-Defined Wireless Sensor Networking (SDWSN) is the paradigm that makes use of a Software-Defined Networking (SDN) approach to the management of Wireless Sensor Networks (WSNs), in an attempt to solve most of the inherent issues with WSNs, for example, resource constraints resulting from the expansion of WSNs [1] [2]. SDWSN results in the separation of the data and control layers of the network which allows for centralized control of the entire network without affecting its overall energy consumption [3].

Within any network, security is one of the most important aspects, and within the SDWSN paradigm, this aspect has had a lack of focus due to the fact that most of the attention has been towards developing the SDWSN architecture itself. Existing work addresses security within SDN [4] - [6] and WSNs [7] - [10] individually, however, there is less research addressing security within SDWSN. The paradigm itself also results in new issues and challenges due to the fact that it is a combination of two developing paradigms.

The fundamental issue with WSN security lies within its implementation. Due to the fact that WSNs consist of small inexpensive sensor nodes with wireless capabilities, they have limited resources, such as processing power. This resource constraints result in the inability to implement WSN security solutions which has therefore led to the development of low resource cryptography methods in order to secure WSNs.

However, one of the advantages of the SDWSN paradigm is that the centralized control results in the freeing up of resources due to the fact that the sensor nodes mainly have to forward data to the controller. Therefore, resource intense cryptographic WSN solutions may be implemented on the control or application planes of the SDWSN architecture.

The two main types of cryptography are symmetric cryptography and asymmetric cryptography. Two basic cryptography methods were implemented within a SDWSN network in order to aid the investigation of the optimal solution regarding cryptography within SDWSN and determine whether the SDWSN paradigm truly results in the ability to implement more resource intense WSN security.

II. WSN Security implementation

The main form of security implementation within WSNs is the use of cryptography in order to secure communications. Although optimal solutions to WSN cryptography are still under investigation, existing solutions may be appropriate when considering their implementation within SDWSN, however there are issues with WSN cryptography as discussed below.

A. WSN Security Implementation Problems

The main issue with regards to WSN cryptography implementation stems from the inherent issues seen within WSNs [7], such as resource constraints, which are significantly increased upon expanding these WSNs. This is due to the fact that most encryption techniques were developed for enterprise/traditional networks and are not feasible for WSNs due to the lack of processing, memory and battery power that they posses [11]. Implementing cryptography requires more processing and additionally may also cause more delays, jitter and packet loss within WSN [7].

Most research focuses on low resource cryptography methods [9], [12], [13] which are primarily separated into symmetric and asymmetric cryptography.

Symmetric cryptography methods are the preferred implementation when it comes to WSNs due to their efficiency

and generally low implementation cost. However, these cryptography solutions result in problems when considering the management of large scale networks. Attempts to improve the scalability of symmetric cryptography methods have come at a cost to component resources. On top of this resistance to scalability, these methods are generally difficult to implement in software.

Asymmetric cryptography methods arose from attempts to mitigate issues with symmetric cryptography such as the simplification of cryptographic key management. The issue with these methods however, is that the methods are too computationally demanding for wireless sensor nodes to handle, therefore running into the fundamental implementation barrier seen within WSNs.

B. Advantages to WSN Security

Although optimal solutions regarding cryptography within WSNs remains a challenge, the nature of the SDWSN paradigm presents a few advantages in this aspect. Due to the centralization of control brought about by the SDN paradigm, the majority of the network programming is handled by a separate controller as opposed to the sensor nodes, thus freeing up network resources which may be used to implement more demanding security solutions.

The cryptography methods discussed above are based on the coupled architecture of the sensor nodes [1] rather than the decoupled nature of the SDWSN architecture, however, these methods and solutions may still be implemented on the control and application plane.

III. WSN Cryptography

In order to secure the transmission of data from the various types of threats present in WSNs, various cryptography techniques are used. These cryptographic techniques are primarily divided between symmetric and asymmetric cryptography as mentioned above.

A. Symmetric Cryptography

The main idea behind symmetric cryptography or secret key cryptography (SKC) is that all the sensor nodes already know secret information (for example the security key itself or information that helps derive the key) before the deployment of the network [12], thus ensuring secure communication throughout the network. This requires the information to be loaded onto the sensor nodes individually before they are deployed within the network.

The overall disadvantage with SKC is that the fact that secret information needs to be loaded onto each node before its deployment makes SKC resistant to scalability and difficulty to implement through software, fundamentally making SKC unfeasible for large scale networks. Additionally, having all the information on all nodes within the network is a security threat as an attacker may be able to compromise the entire network if they compromise a single node as the node will contain secure information that is present on all nodes.

Many different security models have been proposed to mitigate these disadvantages, that make use of different techniques, for example pre-distribution of a random set of keys to each node [14], which was improved to consider deployment knowledge of the pre-distributed keys resulting in the same level of security and connectivity for only a fraction of the previously required keys [15], thus freeing up memory resources within the node. The main disadvantage with this scheme, however is due to the fact and increase in the amount of captured nodes, increases the number of affected links [12].

Alternatively the use of a trusted base station has been proposed that can securely distribute keys as described by Perrig *et al.* [16] set of Security Protocols for Sensor Networks (SPINS) where a trust setup distributes a master key to each node at the time of deployment which is used to derive all other keys. Location-based keys (LBK) has been proposed [17] whereby keys are established and managed using their geographical location. The issue with the use of LBK is that the location information of nodes is not guaranteed within a random deployment [12].

B. Asymmetric Cryptography

The fundamental issue with asymmetric cryptography, or public key cryptography (PKC), is that although it solves most of the issues with SKC, its implementation depends on the use of difficult mathematical problems and therefore the algorithms use more resources, such as energy consumption, than symmetric cryptographic solutions. The main advantage with PKC is the fact that nodes do not need pre-distributed keys, and can establish secure connections using key agreement algorithms or by distributing keys. As is the case with SKC, PKC algorithms have been optimized to mitigate their disadvantages within WSN implementation. It has been stated that both RSA and elliptic curve cryptography (ECC) are suitable for small devices without hardware acceleration [12], and have therefore been the main focus for PKC implementation within WSNs.

The RSA algorithm is named after its developers and is one of the most commonly used PKC algorithms. It allows an encrypted message to be sent without the node having to know a secret key. However, RSA is said to be unfeasible for use within WSNs [18] due to the fact that the 8- or 16-bit processors used in WSNS are unable to provide the necessary memory and processing power to provide strong security. In order to provide strong security, the use of 1024 bit modulus and 160 bit private keys are suggested [19]. However, it has been shown that certain functions of the RSA algorithm can be applied to WSNs [12], as is the case with the TinyPK system [20] which allows authentication and key agreement between sensors that have limited resources. In this system, public RSA functions are implemented on the sensors and private RSA functions are implemented on an external device which results in a promising hybrid solution between SKC and PKC.

ECC outperforms the RSA algorithm on the 8- and 16-bit processors associated with WSNs because of the shorter

transmission messages resulting from the fact that ECC makes use of 160-bit keys over RSA's 1024-bit keys [12] and also due to the faster multiplications seen within ECC as opposed to RSA [21]. Various ECC implementations are discussed each containing different algorithms, optimizations and platforms [12], however, it has been noted that the two popular implementations are TinyECC and ExxM on the TinyOS platform [13].

It is clear that ECC outperforms RSA within WSN implementation, however this may be different when considering SDWSN, where resource limitations are mitigated and it is possible that RSA may be preferred over ECC. The use of PKC has been said to be more advantageous than SKC in terms of both memory usage and complexity, and security resilience [12], which may be emphasized when considering the SDWSN domain.

Additional hybrid solutions between SKC and PKC exist as is the case with McCusker *et al.* [22] who have stated that although SKC systems are appropriate for securing communications between nodes, the fact that keys are pre-installed system wide results in security threats and increased memory consumption, and because of this, are not appropriate. They suggest the use of an asymmetric key system to establish SKC keys between individual nodes. This is achieved using Identity Based Cryptography (IBC) where the identity of the node is used as a public key, instead of using a certificate to bind the nodes identity to a public key [23].

C. Cryptography in SDWSN

When considering the SDWSN paradigm, many WSN cryptography solutions that were thought to be unfeasible can now be adapted and implemented on the control plane thanks to the freeing up of network resources. Additionally, there are developed platforms that can better utilize the decentralization of SDWSN in order to provide an optimal solution with regards to cryptography within SDWSN.

The idea of a trusted base station as shown in the SPINS platform [12] is an example this. The SDWSN paradigm presents a major advantage when considering the fact that the platform relies on a trusted base station to distribute master keys. Due to the centralization of control and security management brought about by SDWSN [24], the controller can act as the trusted base station and can therefore be enforced with SPINS security kernal in order to provide a unique solution to cryptography within SDWSN.

With regards to PKC, more resources can be allocated into implementing RSA cryptography, which is considered the standard, within the controller. Additionally some parts of the algorithm can be applied to the nodes as outlined in the TinyPK system. Within WSN it is evident that the smaller ECC algorithm is the best PKC technique, however as stated, this may not be the case within SDWSN and therefore more work must be done to confirm whether this is the case.

IV. Cryptographic Algorithm Comparison

In order to determine whether SKC or PKC is the best solution for cryptography within SDWSN, the most popular SKC and PKC algorithms are described and implemented within an SDWSN network. The two selected algorithms are the AES algorithm for SKC and the RSA algorithm for PKC, and are chosen due to their ease of implementation and popularity. The implementation of these algorithms are explained below.

A. Symmetric Cryptography: AES Algorithm

The AES encryption algorithm is a symmetric block cipher that is able to process data block inputs of 128 bits using keys of length 128, 192 and 256 bits. Four different transformations are used in order to encrypt a data input is shown below (this is known as the cipher) [25].

- Byte Substitution using a substitution table (S-box).
- Shifting rows of state array by different offsets.
- Mixing data within columns of the state array.
- Adding a round key to the state.

The process of encryption is shown in the block diagram in Fig. 1.

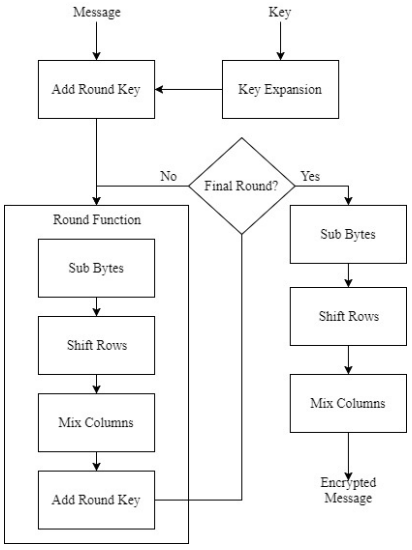

Fig. 1. AES Cipher block diagram

To begin the encryption the message is copied to a state array. Thereafter an initial round key is added and then the state array is transformed using a round function. A round function is a function that applies the four transformations described in the order they are described. Depending on the key size, the round function can be implemented 10, 12 or 14 times with the final round being different in that it does not apply the mix columns transformation. Key expansion refers to the function that generates a key schedule from the given key. Thereafter, to decrypt the ciphertext, all cipher transformations are inverted in the reverse order, producing an inverse cipher.

B. Asymmetric Cryptography: RSA Algorithm

The RSA algorithm, introduced in 1978, is named after its developers and is one of the most popular forms of PKC. Not only does it implement a PKC, but it also implements digital

signatures. The idea behind PKC is to have a public encryption key and a private decryption key, therefore only the correct decryption key is able to decrypt the message [21]. Due to the fact that the encryption keys are public, both keys must be produced in a way so that the decryption key is not easily deduced. In order to verify the origin of the message, digital signatures are used in the decryption key.

The algorithm uses two prime numbers to create a large prime number which is then used to encrypt and decrypt a message [26]. In order to create a secure key and encryption scheme the process requires large mathematical functions and many multiplications which is why the RSA algorithm has been previously too resource intense to implement within WSNs.

V. Algorithm Simulation

In order to verify which algorithm works better within an SDWSN network, the two algorithms were implemented using the IT-SDN tool [27]. IT-SDN is an open source SDWSN tool inspired by TinySDN, however it is independent of operating systems and its functions. It primarily makes use of Contiki and the simulations themselves were done using the Cooja tool. The two algorithms were implemented in the same manner as depicted in Fig 2.

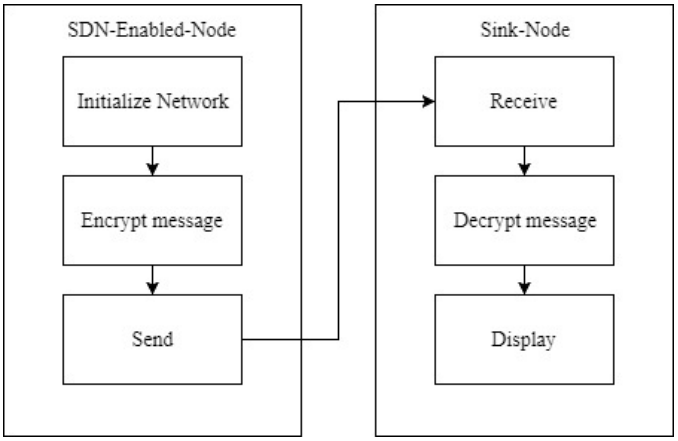

Fig. 2. Simulation block diagram

Firstly, the network is initialized and all nodes are connected to the controller. For the simulation, three SDN-enabled nodes were used with one controller and one sink node. The same layout was used for three different simulations. One simulation where no encryption was implemented, then two more each implementing AES encryption and RSA encryption. The SDWSN nodes are programmed to send a message every 60 seconds. These messages are then received by the sink node. Encryption was carried out on the nodes to encrypt the message before it was sent. This encrypted message was then received by the sink node and decrypted before it was displayed to give a visual indication as to whether the decrypted message was the same as the original message.

All three simulations were carried out over a time of 20 minutes and using the same platform. The sensor node platform was originally the Arago CC2520 wismote, however these motes were unable to compile the required encryption. Therefore these motes were replaced with the Tmote sky low power wireless sensor module [28]. It is important to note the sensor node platform used as it to calculate the power consumption of the network simulation.

A. Simulation metrics

In order to verify if the implemented algorithm performed as required and fundamentally determine which algorithm was better, the following simulation metrics were monitored.

- Energest CPU, LPM, TX and RX values
- Overall delivery rate
- Packet data ratio
- Overall delay

In order to obtain these metrics, a Qt based monitoring interface was used. This interface makes use of IT-SDN's statistic script to obtain statistical data such as overall delivery rate, and an Energest script in order to obtain energy values. Energest Power profile is a set of equations that makes use of power metrics derived from Contiki's Powertrace application. In order to calculate the power consumption, Energest makes use of the following equation.

$$Power = \frac{EnergestValue \times Current \times Voltage}{RtimerSecond \times Runtime} \quad (1)$$

Where the energest value is the value CPU, LPM, TX and RX value provided by powertrace, RTIMER_SECOND is the number of ticks per second, runtime is the interval at which the measurements are taken and the current and voltage values are the values derived from the datasheet of the Tmote Sky nodes.

The other statistical data, such as overall delivery rate, will provide an indication as to the success of the encryption implementation. A low delivery rate and packet data ratio will correspond with data packets failing to deliver. The overall delay will provide another indication as to which encryption implementation is more efficient in terms of time taken to process the algorithm.

B. Results

The results obtained from the simulation of the SDWSN network are shown in TABLE I and Fig. 3. TABLE I contains the main power measurements and statistical data for the simulation and Fig. 3 shows the average power usage of the simulation.

The main power measurements in TABLE I are average power and total power. The average power measurement is the average power for all nodes in the simulation for each of the three simulations. The total power measurement is the sum off all the obtained power values for all nodes for each of the three simulations. As shown in TABLE I, the average and total power values are similar for the simulation of no encryption and AES encryption (with a difference of 100 μW and 92.7 mW respectively), whereas in the case of the RSA

encryption simulation the average and total power values are slightly larger with a difference of 3.3 mW and 257.3 mW respectively.

The statistical data captured from each simulation is an indication as to the success of the encryption implementation within the network. The overall delivery rate is the ratio between all packets delivered and all packets received, the data packet fraction is the ratio of data packets to other packets (such as control set up packets) and the overall delay is the average delay for all packets in the simulation. All three simulations have a delivery rate of 100% and a data packet fraction of 65.93%. However, the overall delay is longer by 3.42 ms for AES Encryption and 6.67 ms for RSA encryption.

Fig. 3 shows the average power over the simulation time for all nodes within the network. This figure is a visual indication of the fact that the simulated RSA encryption resulted in higher energy values overall.

VI. Discussion and Remaining Challenges

As shown by the results, the implementation of both the AES and RSA encryption algorithms using IT-SDN was successful. This is shown by the overall delivery rates and data packet fractions in TABLE I. The fact that the delivery rates are 100% and the data packet fractions are all the same shows that the algorithm implementation was successful as all packets were sent successfully and additionally, the data packet fraction of 65.93% shows that most of the packets sent in the network were data packets.

With regards to the energy metrics, it can be seen that RSA encryption consumes more power within the simulation as opposed to that of AES encryption. This is due to the fact that the RSA algorithm is slightly more computationally intense than that of AES. Therefore in terms of energy consumption, the AES algorithm is better, thus providing an initial indication as to which cryptographic method is better. Additionally, it is shown that the overall delay is lower for AES encryption than RSA encryption. This is another indication that the AES algorithm is better than the RSA algorithm with regards to its implementation within SDWSN.

However, the difference in performance and energy usage is not significantly large as expected for the RSA algorithm and although the AES algorithm may be slightly more energy efficient and faster, this is not an indication as to the overall security gained by using a specific algorithm. It is important to note that no attempts to break into the network were made due to the fact that a comparison between SKC and PKC with regards to the overall resource and energy consumption was the main purpose of this paper. Therefore, although the AES algorithm outperforms the RSA algorithm, the RSA may provide better security overall. Thus there is fundamentally a slight trade-off between security and efficiency, which when considering the fact that the RSA algorithm consumes slightly more energy, may work in favor of RSA.

It is also important to note, that only the two simplest algorithms were chosen in order to implement these algorithms on the IT-SDN platform. Many other implementations exist for platforms such as TinySDN and SDN-Wise, however not for IT-SDN, and therefore the two simplest, most popular algorithms were chosen to be implemented. Therefore, using other PKC algorithms such as ECC should yield better results with regards to energy efficiency and overall security. Additionally, due to the fundamental nature of SDWSN, the RSA algorithm can also be implemented within the network as shown in the results above. Therefore, solutions based on the RSA algorithm can now be considered for use in SDWSN.

However, the same could be said for SKC algorithms. It has been shown that SKC key management can be carried out through the use of a trusted base station as is the case with SPINS. This idea sounds very promising when considering its possible implementation within SDWSN. The SDWSN controller can act as a base station and could be used in conjunction with a security kernel or management interface in order to carry out key management. This idea must still be tested within the SDWSN paradigm, but could provide a promising solution to cryptography within SDWSN. Additionally, hybrid solutions such as the one presented by McCusker *et al.* [22] may prove advantageous when considering their effect within SDWSNN.

The results above essentially prove that AES encryption, and by extension SKC cryptography is less resource intensive than RSA encryption when implemented within an SDWSN network, however this may not be true for all SKC cryptography methods, for example, ECC cryptography. Additionally, hybrid methods and methods that consider the nature of SDWSN itself may prove a better solution for cryptography within SDWSN.

VII. Conclusion

Various simulations were carried out that test the implementation of both the AES algorithm and RSA algorithm within the IT-SDN platform. The results show that the algorithms were successfully implemented within an SDWSN network, and that the AES algorithm is the more efficient algorithm in terms of resource usage and energy consumption.

However, this does not fundamentally answer the question as to which cryptography method is better within SDWSN. Although the AES algorithm performs better within the network, there is no guarantee that the security provided by this encryption is sufficient for the network. Therefore, the use of PKC methods such as RSA and ECC may prove to have a better trade-off between security and efficiency.

Additionally, new platforms could be developed that consider the nature of SDWSN. These platforms can be based on the work done by Perrig *et al.* [16] and combine both symmetric and asymmetric solutions in order to provide a true clear solution for cryptography within SDWSN.

References

[1] H. I. Kobo, A. M. Abu-Mahfouz, and G. P. Hancke, "A survey on software-defined wireless sensor networks: Challenges and design requirements," *IEEE Access*, vol. 5, pp. 1872–1899, Feb. 2017.

[2] M. Ndiaye, G. P. Hancke, and A. M. Abu-Mahfouz, "Software defined networking for improved wireless sensor network management: A survey," *Sensors*, vol. 17, no. 5:1031, pp. 1–32, 2017.

TABLE I
SIMULATION RESULTS

	Average Power (mW)	Total Power (mW)	Delivery Rate (%)	Overall Delay (ms)	Data Packet Fraction (%)
No Encryption	1.0943	87.5468	100	498.08	65.93
AES Encryption	1.0944	87.5495	100	501.5	65.93
RSA Encryption	1.0976	87.8041	100	504.75	65.93

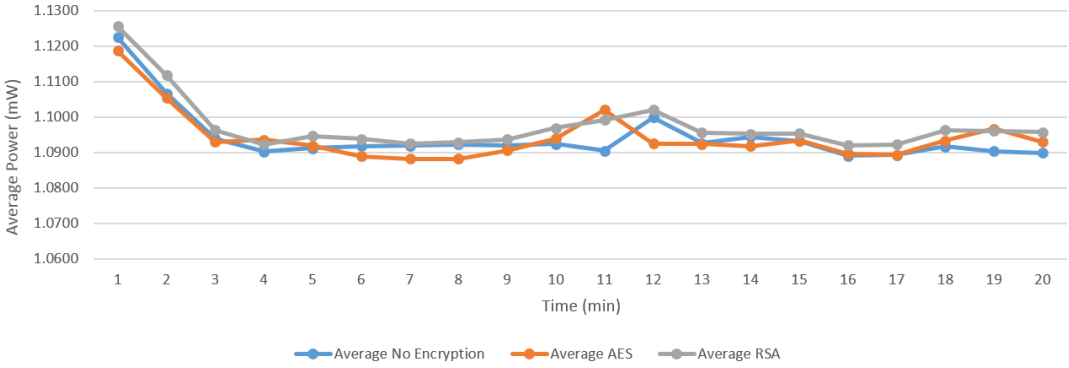

Fig. 3. Average Power over Time

[3] K. M. Modieginyane, B. B. Letswamotse, R. Malekian, and A. M. Abu-Mahfouz. (2017, Mar.) Software defined wireless sensor networks application opportunities for efficient network management: A survey. Computers and Electrical Engineering. [Online]. Available: http://dx.doi.org/10.1016/j.compeleceng.2017.02.026

[4] D. Kreutz, F. M. V. Ramos, and P. Verissimo, "Towards secure and dependable software-defined networks," in *Proc. Second ACM SIGCOMM Work. Hot Top. Softw. Defin. Netw. - HotSDN 13*, 2013, p. 55.

[5] S. T. Ali, V. Sivaraman, A. Radford, and S. Jha, "A survey of securing networks using software defined networking," *IEEE Trans. Rel.*, vol. 64, no. 3, pp. 1086–1097, Sep. 2015.

[6] I. Ahmad, S. Namal, M. Ylianttila, and A. Gurtov, "Security in software defined networks: A survey," *IEEE Commun. Surveys Tuts.*, vol. 17, no. 4, pp. 2317–2346, Aug. 2015.

[7] A. S. K. Pathan, H. W. Lee, and C. S. Hong, "Security in wireless sensor networks: issues and challenges," in *2006 8th Int. Conf. Adv. Comm. Tech.*, vol. 2, 2006, p. 1048.

[8] H. Modares, R. Salleh, and A. Moravejosharieh, "Overview of security issues in wireless sensor networks," in *2011 Third Int. Conf. Comput. Intell. Model. Simul.*, 2011, pp. 308–311.

[9] T. Zia and A. Zomaya, "Security issues in wireless sensor networks," in *2006 International Conference on Systems and Networks Communications (ICSNC06)*, 2006, p. 40.

[10] A. M. Abu-Mahfouz and G. P. Hancke, "Evaluating ALWadHA for providing secure localisation for wireless sensor networks," in *Proc. IEEE AFRICON 2013 conf.*, Mauritius, Sep. 2013, pp. 501–505.

[11] J. Yick, B. Mukherjee, and D. Ghosal, "Wireless sensor network survey," *Computer Networks*, vol. 52, no. 12, pp. 2292–2330, Aug. 2008.

[12] K. Chelli, "Security issues in wireless sensor networks: Attacks and countermeasures," in *Proc. World Congr. Eng.*, vol. 1, 2015.

[13] J. Louw, G. Niezen, T. D. Ramotsoela, and A. M. Abu-Mahfouz, "A key distribution scheme using elliptic curve cryptography in wireless sensor networks," in *Industrial Informatics (INDIN), 2016 IEEE 14th International Conference*, 2016, pp. 1166–1170.

[14] H. Chan, A. Perrig, and D. Song, "Random key predistribution schemes for sensor networks," in *Proc. Sym. on Security and Privacy*, Berkeley, USA, May 2003, pp. 197–213.

[15] W. Du, J. Deng, Y. S. Han, S. Chen, and P. K. Varshney, "A key management scheme for wireless sensor networks using deployment knowledge," in *Proc. 23rd IEEE Comp. and Comm. Soc. - INFOCOM 2004*, Hong Kong, China, Mar. 2004.

[16] A. Perrig, R. Szewczyk, J. D. Tygar, V. Wen, and D. E. Culler, "Spins: Security protocols for sensor networks," *Wireless Networks*, vol. 8, no. 5, pp. 521–534, Sep. 2002.

[17] D. Liu and P. Ning, "Location-based pairwise key establishments for static sensor networks," in *Proc. 1st ACM Work. Sec. Ad Hoc Sen. Netw - CCS'03*, Oct. 2003, pp. 72–82.

[18] A. Perrig, R. Szewczyk, J. Tygar, V. Wen, and D. Culler, "SPINS: Security protocols for sensor networks," *Wireless Networks*, vol. 8, no. 5, pp. 521–534, Sep. 2002.

[19] National Institute of Standards and Technology. (2007, Mar.) NIST Computer Security Resource Center. [Online]. Available: $csrc.nist.gov/publications/nistpubs/800-56A/SP800-56A_Revision1_Mar08-2007.pdf$

[20] R. Watro, D. Kong, S. fen Cuti, C. Gardiner, C. Lynn, and P. Kruus, "TinyPK: Securing Sensor Networks with Public Key Technology," in *Proc. 2st ACM Work. Sec. Ad Hoc Sen. Netw - SASN'04*, Oct. 2004, pp. 59–64.

[21] N. Gura, A. Patel, A. Wander, H. Eberle, and S. C. Shantz. Comparing elliptic curve cryptography and rsa on 8-bit cpus. Sun Microsystems Laboratories. [Online]. Available: http://www.research.sun.com/projects/crypto

[22] K. McCuster and N. O'Connor, "Low-energy symmetric key distribution in wireless sensor networks," *IEEE Transactions on Dependable and Secure Computing*, vol. 8, no. 3, pp. 363–376, 2011.

[23] A. Shamir, "Identity-Based Cryptosystems and Signature Schemes," in *Proc. Crypto '84*, Aug. 1984, pp. 47–54.

[24] S. W. Pritchard, G. P. Hancke, and A. M. Abu-Mahfouz, "Security in software-defined wireless sensor networks: Threats, challenges and potential solutions," in *IEEE International Conference of Industrial Informatics, In Press*, Emden, Germany, Jul. 2017.

[25] F. PUB, *Announcing the ADVANCED ENCRYPTION STANDARD (AES)*, FIPS FIPS PUB 197, 2001.

[26] R. Yashaswini, H. G. Nayana, and B. A. Thomas, "Wireless sensor network security using cryptography," *International Journal of Advanced Research in Computer Science and Technology (IJARCST 2016)*, vol. 4, no. 2, pp. 185–189, Jun. 2016.

[27] R. C. A. Alves, D. Oliveira, G. Nez, and C. M. S. de Ferramentas. It-sdn: Improved architecture for sdwsn. Larc USP. [Online]. Available: http://www.larc.usp.br/users/cbmargi/www//it-sdn/

[28] *Tmote Sky: Ultra low power IEEE 802.15.4 compliant wireless sensor module*, Moteiv Corporation, Jun. 2006.