Performance Analysis of Cryptography Algorithms in Wireless Sensor Network Based on Key Strength and Data Acceptance Probabilities

Mohammed Alwakeel[*1, 2], **Sultan Swailem Alatawi**[1]

[1]Faculty of Computers and Information Technology, University of Tabuk, Tabuk, Saudi Arabia

[2]Sensor Networks and Cellular Systems (SNCS) Research Center , Tabuk, Saudi Arabia

[*] Corresponding Author: *alwakeel1[at]gmail.com*

Abstract: ***A multitude of fields and applications now make use of wireless sensor networks. Yet, despite their versatility and usefulness, these types of networks come with two major drawbacks: Firstly, due to the nature of the fact that they are wireless, this type of networks are very susceptible to attacks and intruders. The second major issue concerns that the limited and irreplaceable resources used in these networks do not generally allow for heavy computation, secure data packing, and sender authentications over an extended period. For these reasons, the selection of a suitable cryptography algorithm that provides an acceptable security level while keeping the use of network resources to a minimum is a task of vital importance. In this paper, the development of a model that allows for the comparison and evaluation of the encryption algorithms used in wireless sensor networks is presented. The model proposed assesses algorithms along two metrics: key strength probability and data acceptance probability. The results from the evaluation of the model using a simulation are encouraging, and we suggest that the model we propose will aid network administrators to select the most appropriate encryption algorithm, namely that which exceeds the desired security level for their network while simultaneously requiring the lowest amount of network resources possible.***

Keywords: Wireless sensor network; Cryptography Algorithm; Key Strength Probability, Data Acceptance Probability.

1. Introduction

Wireless sensor networks (WSNs) consist of many devices distributed in multiple locations to collect real-time information. WSNs are commonly used in various fields, including health care, smart homes, intelligent transportation and many other applications [1], [2], [3], [4]. The sensor nodes in WSNs are designed to collect data from the surrounding environment and then relay this data to a base station. Since there are different types of WNS-nodes in use, the exact method of data exchange between the nodes of a WSN nodes and its base station can vary considerably. However, regardless of data-exchange method, the confidentiality of the data must be constantly maintained [5].

When WSNs nodes are spread over a wide geographical area, the networks are threatened by several types of attacks: security might be compromised if an attacker plants a node into the network, or a security flaw in the network is exploited and the attack then eavesdrops on seemingly secure data. However, there are several security solutions may be applied to increase the security of networks. In addition, several cryptography techniques are proposed to secure the communications between the elements in a WSN environment. However, compared to conventional wired and wireless networks, the selection process and application of a security algorithm within an WSN is rather complex, since not all encryption methods are useable in all types of WSNs, as the exact limitation of each WSN is determined by the nature and capabilities of the sensor nodes that are being used [5].

Many cryptography algorithms can encrypt the data, but each of these come with their own pros and cons. Algorithms use encryption keys to complete the encryption and decryption process [6], in general, cryptography algorithms can be sorted into three classes: Symmetric Key Cryptography (secret or private key cryptography), Asymmetric Key Cryptography (public key), and Hash function, each of these classes holds a number of algorithms and techniques.

A single key is used in symmetric cryptography for encryption and decryption; examples for this type are DES, 3DES, AES, RC4, 3DES, and AES [7]. A single key is easy to implement in small nodes, but issues may arise due to continuous key management and frequent secure exchange. The asymmetric technique makes use of public-private keys pairs for data encryption and decryption respectively, and an example algorithm is RSA [7], [8], as it requires very advanced mathematical operations, this second type of encryption is slower. Hash-based encryption converts all messages to a fixed-length cipher. MD2, MD5, and SHA are the most common examples of this type, which is frequently used for the encryption of user passwords. Plain text is assigned to the hash function as a parameter, and the hash function then encrypts this plain text into a text with a fixed-size.

The continuous development of tools and new technologies causes attackers and security experts to engage in a constant race against each other. Attackers constantly crack codes and attempt breaches, which leads to the development and further improvement of security protocols.

Server parameters group different encryption techniques together according to different characteristics, such as:

- Encryption and decryption key-type (symmetric key or asymmetric key)
- key-length used to determine cryptographic system-strength
- Number of potential steps of an attack to the cryptographic system
- Potential attack time for a system breach

A number of methods exist to evaluate the performance of a specific technique of algorithm, or a group of algorithms, and a number of parameters may be chosen for evaluation. One article [9] evaluated a number of different algorithms, using processing time as a parameters, Table 1 below shows a comparison of how many resources are required by different algorithms to process information.

Table 1: Comparison of Memory Usage

Algorithm	Memory used (KB)	The average number of bits required to encode one byte of data
DES	18.2	27
3DES	20.7	40
AES	14.7	256
Blowfish	9.38	128
RSA	31.5	44

2. Encryption Algorithms Performance Metrics

Since heavy processing and frequent key exchanges are extremely difficult to achieve with WSN, the selection of the proper encryption algorithm and key management mechanism is essential. Key management techniques for WSNs are generally evaluated by three metrics: Security, Efficiency, and Flexibility [10]. Each of these three metrics contains several parameters for evaluation. Security metrics include three parameters: node authentication, Resilience, and Revocation, the metrics for efficiency comprises of five parameters, namely memory needed, computing required, the bandwidth required, energy consumption, and secure connectivity, while the flexibility parameters comprise of two parameters, namely lack of prior deployment knowledge (LPDK) and scalability. LPDK describes the ability of nodes to create keys and transfer data between nodes whose exact location has not been previously established. Scalability is a measurement for the networks ability to add nodes to the network without compromising the security of the network while nodes are being added.

The authors of research [11] evaluated six different symmetric algorithms, namely AES, DES, 3DES, RC2, Blowfish, and RC6 according to a number of criteria, including encoding and decoding speed, power consumption, and key size. This simulation-based research found that the DES algorithm is more suitable for use in WSNs than 3DES, and that additionally, RC2, RC6, and Blowfish require higher amounts of energy than the other algorithms, particularly when having to handle image files. In the case of AES and RC6 algorithms, it was found that node power consumption is related to the size of the encryption key. A different article [12] investigated the safety of cloud computing and investigated the performance of three different types of security algorithms RSA, MD5, and AES, the results of this comparison are provided below in Table 2.

Table 2: Comparison of the mean processing time of the three algorithms on the cloud and a single processor (local) for different input sizes

Input size	RSA (Local)	RSA (Cloud)	MD5 (Local)	MD5 (Cloud)	AES (Local)	AES (Cloud)
2 Kb	678.4	380.2	15.6	0.7	425	2.3
5 Kb	747.3	390.2	15.9	0.9	445.7	8.2
10 Kb	796.8	400.9	15.9	1	454.2	15.5
20 Kb	853.4	429	16	1.4	487.4	24.8

A further study [13] compared the DES, AES, and RSA algorithms regarding their computation time, memory usages, and output, and results indicate that the RSA algorithm took more time to complete the encryption task than the other two algorithms. Additionally, it was found that the AES algorithm had the lowest memory usage, while the RSA algorithm's memory usage was the highest. An additional article [14] evaluated the different encryption algorithms used to encrypt video files. Video files of various sizes and formats were used to investigate the encryption and decryption times required by the different algorithms. It was found that the AES algorithm was faster than the other two algorithms (DES and Blowfish). A further article [15] extended upon this study and evaluated DES, 3DES, and AES algorithms against nine different factors; the results are shown in Table 3.

Table 3: Comparison between AES, 3DES and 3DES.

Factors	AES	3DES	DES
Key length	128, 192, 256 bits	112 - 168 bits	56 bits
Cipher type	Symmetric	Symmetric	Symmetric
Block size	128, 192, 256 bits	64 bits	64 bits
Developed	2000	1978	1977
Cryptanalysis resistance	strong against differential, truncated differential, linear, interpolation, and square attacks	vulnerable to differential Brute Force attacker could be analyzed plaint text using differential cryptanalysis	vulnerable to differential and linear cryptanalysis; weak substitution tables
Security	Considered secure	One only weak which exists in DES	Proven inadequate
Possible keys	2^{128}, 2^{192}, 2^{256}	2^{112} or 2^{168}	2^{56}
Possible ASCII printable	95^{16}, 95^{24}, 95^{32}	95^{14}, 95^{21}	95^{7}

character keys			
Time required to check all possible keys at 50 billion keys per second	for 128 bit key: 5X 10^{21} years	for 112 bit key: 800 days	For a 56 bit key: 400 days

3. Proposed Performance Analysis Model for Encryption Algorithms

To ensure the safety of the data transmitted via conventional and wireless sensor networks, a number of encryption algorithms are in use. As already mentioned above, many metrics may be used to evaluate the performance of these algorithms, including complexity, capabilities, level of provided security, and the resources required for operation. Due to the limitations of WSN, the algorithm that provides the highest level of security may not be the optimal choice, instead, an algorithm that requires fewer network resources while simultaneously offering an adequately high level of protection is ideal. Network administrators therefore need to consider keeping a balance between utilization of resources and the required security level for the network.

The model presented in this research was developed with the aim to provide WSN administrators with a useful tool to compare and contrast several encryption algorithms based on the parameters of key strength probability and reading dropping probability. Administrators will thus be able to identify the encryption algorithm which, one the one hand, meets or exceeds the desired security level for their network and makes uses of the minimum amount of resources, while on the other hand also requires the smallest amount of network resources. Reading dropping probability may be defined as the probability that reading from a sensor will be dropped during transmission from a sensor to the base station if the strength of security is insufficient. The present article therefore contributes to existing knowledge in two ways: One, by defining parameters and metrics that may be used in the evaluation of any encryption algorithm used in specific WSNs. The second contribution is the development of a model to compare between different encryption- and key management-techniques. The model can be utilized to identify the most suitable algorithm for a specific WSN environment.

Our simulation model compares encryption algorithms based on their key strength probability (KSP). KSP may be defined as the probability that a sensor's cryptographic key is not discovered during a brute force attack after a period of time (t) [16]. Additionally, reading dropping probability (P_{RD}) is also considered within the model's evaluation process. P_{RD} may be defined as the probability that a reading from a sensor within the network is dropped (i.e., rejected) at a sensor as a result of an insufficient key strength level.

The data is accepted from the sensor only if the KSP probability is greater than or equal to a certain threshold $Pr\ (KSP \le th_{ksp})$ as the security level is sufficient. In addition, reading dropping probability (P_{RD}) and data acceptance probability (P_{DA}) for an algorithm is computed as:

$$P_{DA} = 1 - P_{RD} \qquad (1)$$

The KSP as defined above is the complement of a successful brute force attack after the time (t):

$$KSP = 1 - P_{bf} \qquad (2)$$

$P_{bf}(t)$ refers to the probability of a successful brute force attack after time passed (t), which starts when a key was activated by a node for encryption, where [16]:

$$P_{bf}(t) = \frac{N(t)}{2^s} \qquad (3)$$

$N(t)$ is the number of keys an attacker try until time (t) has elapsed, while s denotes the strength of the encryption algorithm (in bits), then $N(t)$ can be established by multiplying the time (t) by the number of keys that an attacker can try per time unit (k), then

$$N(t) = k \times t \qquad (4)$$

From the definition of KSP and the equations (2), (3), and (4), we get [16]

$$KSP = \begin{cases} 1 - \frac{Kt}{2^s} t \le \frac{2^s}{K} \\ 0 \qquad\qquad\qquad else \end{cases} \qquad (5)$$

4. The Simulation Model

There are two ways to trigger a sensor node: either via an activation mechanism or receiving data from a neighboring sensor, which in turn activates the routing protocol to send the data onwards to its destination. For the design of our simulation model, the following was assumed:

- The sensors used are identical.
- All sensors have been placed in predetermined positions (deterministic deployment scheme)
- The routing protocol that ensures that data is transmitted from a sensor to the base station will use the shortest route possible.
- The network is free of congestion.
- Within the network, any sensor can communicate with any sensor that is adjacent
- A sensor will be selected randomly based on uniform distribution random process

Given these assumptions, the value for the time (t) in equation (5) is independent at each sensor, and may be approximated using gamma distributed random variable with probability density function defined as:

$$f(t) = \frac{\beta^{\alpha}}{\Gamma(\alpha)} t^{\alpha-1} e^{-\beta t} \qquad (6)$$

In this function, α is the shape parameter, β is the rate parameter and 1/β is the scale parameter), and Γ(x) is defined as

$$\Gamma(x) = \int_0^{\infty} y^{x-1}\, e^{-y} dy \qquad (7)$$

As outlined earlier, the simulation model allows for the will be used to comparison of several encryption algorithms based on their data acceptance probability (P_{DA}) and reading dropping probability (P_{RD}). It is worth pointing out that the parameters α and β of equation (6) also affect approximate (t) accuracy. Methods as to how additional (t) accuracy can be established may be developed in later research. The parameters (k) and (s) in equation (5) depend on the encryption algorithm that is being analyzed.

5. Results and Discussion

The present research introduces a simulation model for the comparison of different encryption algorithms, using P_{DA} as the parameter. The minimum required value for P_{DA} is denoted by a threshold (th_{PDA}). Figure 1 shows the data acceptance probability vs. KSP threshold. As it can be seen from this figure, the data acceptance probability decreases as the threshold increases. Additionally, it can be seen that by setting a particular data acceptance probability threshold and KSP probability threshold, the algorithms that satisfy the desired level of security can be identified. To illustrate, if the data acceptance probability threshold = 0.75 and the desired KSP probability threshold = 0.79, then an algorithm with a strength of less than (s=56) cannot be recommended for use.

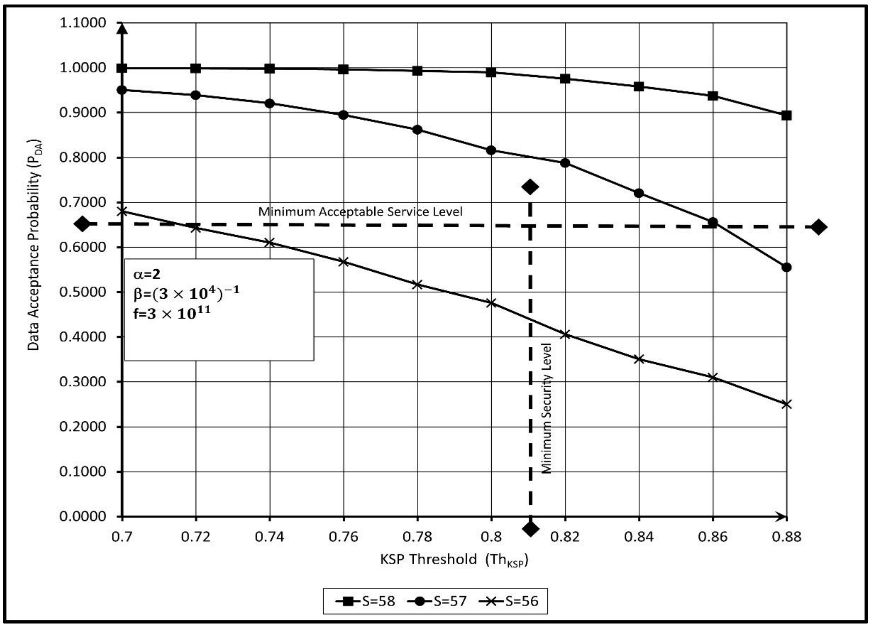

Figure 1: Data Acceptance Probability vs. KSP Threshold

Figures 2 and 3 show the effect of changing the scale parameter and the shape parameter. As the figure demonstrates, increasing the scale parameter leads to a decrease data acceptance probability. This is expected behavior, as increase the scale parameter means an increase of (t). Changing the parameter has a direct impact of the distribution of (t), as well as the mean and the variance of (t). Through additional and more elaborate analysis as to how these parameters may be selected, future works may build upon this observation.

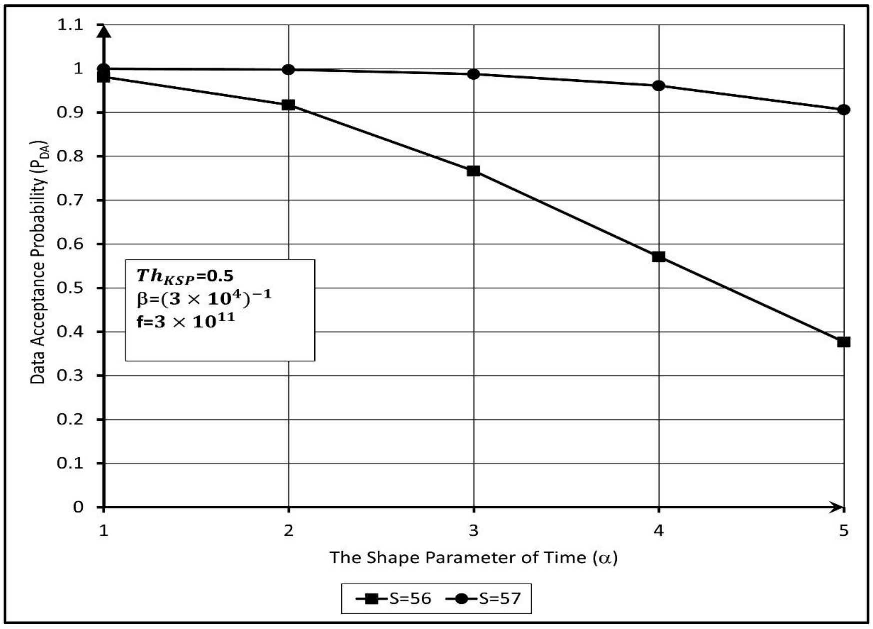

Figure 2: Data Acceptance Probability vs. Shape Parameter of Time (α)

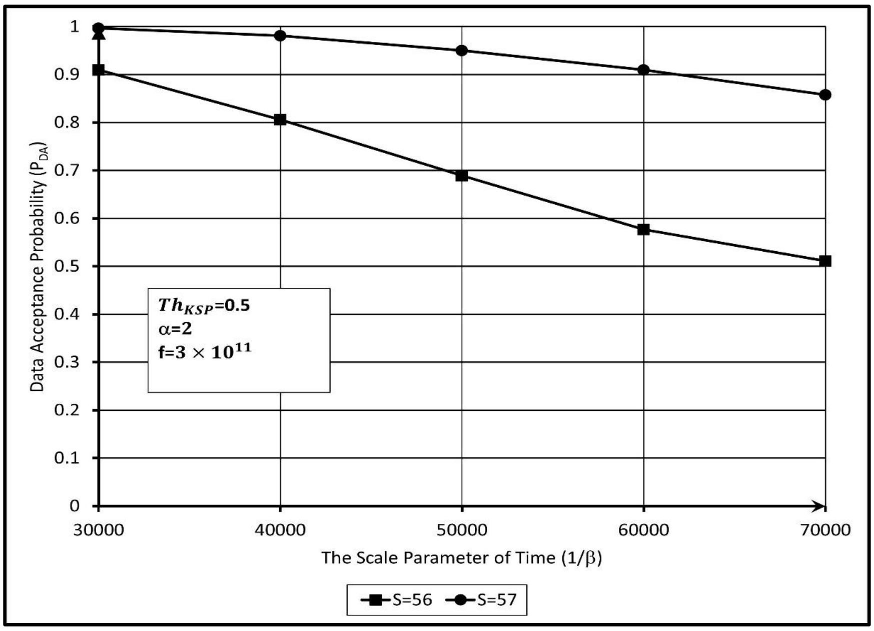

Figure 3: Data Acceptance Probability vs. Scale Parameter of Time (1/β)

6. Conclusion

A number of different algorithms are used to protect the data within wireless sensor networks. Choosing a specific algorithm can be a difficult task, as thorough and careful analysis is required to determine the algorithm most appropriate for the specifications of a specific WSN. The main contribution of the present paper is that it presents a model that allows for the comparison and evaluation of the performance of several cryptography algorithms. The present paper also uses data acceptance probability as an additional metric, which is an essential contribution to the overall research in this area. The system is evaluated using a simulation model. Several varying simulation runs were performed and different encryption algorithms are tested in based on their data acceptance probability. Simulation results suggest that it the algorithm with the highest strength-level of security is not always the most suitable solution, but that if the characteristics of a WSN are taken into account, then ideal choice is an algorithm that makes the most efficient use of resources while offering the optimum security is preferable. In the model that is introduced in this paper, gamma distributed random variables have been used to approximate the time. For future research, it is recommended that techniques such as Maximum Likelihood Estimation and Maximum A Posteriori to determine the parameters of the random variable (t) (namely α and β) are investigated, as this would help to establish the time (t) in the simulation model more accurately.

7. Acknowledgment

The authors gratefully acknowledge the support of the SNCS Research Center at the University of Tabuk, Saudi Arabia.

References

[1] O. Olakanmi, A. Dada. Wireless Sensor Networks (WSNs): Security and Privacy Issues and Solutions. In *Wireless Mesh Networks - Security, Architectures and Protocols,* London, UK. Intech Open Limited, **2020**.

[2] R. Agarwal, R.V. Martinez-Catala, S. Harte, C. Segard, B. O'Flynn. Modeling Power in Multi-functionality Sensor Network Applications. In 2008 Second International Conference on Sensor Technologies and Applications, **2008**; pp. 507–512.

[3] D. Benhaddou, A. Al-Fuqaha. Wireless Sensor and Mobile Ad-Hoc Networks: Vehicular and Space Applications. New York, NY.: Springer New York, **2015**.

[4] D. Kandris, C. Nakas, D. Vomvas, G. Koulouras. Applications of Wireless Sensor Networks: An Up-to-Date Survey, *Appl. Syst. Innov*.**2020**, vol. 3, no. 1, p. 14.doi:10.3390/asi3010014

[5] C. Lee. Security and Privacy in Wireless Sensor Networks: Advances and Challenges. *Sensors*, **2020**; vol. 20, no. 3, p. 744. **https://doi.org/10.3390/s20030744**

[6] W. Stallings. Cryptography and network security, Principles and practices. 6th ed. Saddle River, NJ.: Prentice Hall, **2005**.

[7] W. Wardlaw. The RSA Public Key Cryptosystem. In Coding Theory and Cryptography, Berlin, Heidelberg: Springer Berlin Heidelberg, **2010**; pp. 101–123.

[8] R. Rivest, A. Shamir, L. Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Communication of the ACM,* **1978**; vol. 21, no. 2, pp. 120–126.

[9] M. Wahid, A. Ali, B. Esparham, M. Marwan. A Comparison of Cryptographic Algorithms: DES, 3DES, AES, RSA and Blowfish for Guessing Attacks Prevention. *J. comput. sci. inf. tech*.**2018**; vol. 3, no. 2, pp. 218–230.

[10] M. Simplício, P. Barreto, C. Margi, T. Carvalho. A survey on key management mechanisms for distributed

Wireless Sensor Networks. *Computer Networks*, **2010**; vol. 54, no. 15, pp. 2591–2612.

[11] R. Rivest, A. Shamir, L. Adleman. A method for obtaining digital signatures and public-key *cryptosystems*. Communication of the ACM, **1978**; vol. 21, no. 2, pp. 120–126.

[12] P. Arora, A. Singh, H. Tyagi. Evaluation and Comparison of Security Issues on Cloud Computing Environment. *World of Computer Science and Information Technology Journal*, **2012**; vol. 2, no. 5, pp. 179–183.

[13] S. Seth, R. Mishra. Comparative Analysis of Encryption Algorithms for Data Communication. *Int. j. comput. sci.***2011**; vol. 2, no. 2, pp. 292–294.

[14] S. Pavithra, E. Ramadevi. Performance Evaluation of Symmetric Algorithms. *J. glob. res. comput. sci. technol.***2012**; vol. 3, pp. 44–45.

[15] H. Alanazi, B. Zaidan, A. Zaidan, H. Jalab, M. Shabbir, Y. Al-Nabhani. New Comparative Study between DES, 3DES and AES within Nine Factors. *J. Comput.***2010**; vol. 2, no. 3, pp. 152–158.

[16] A. Ramos, R. Filho. Sensor Data Security Level Estimation Scheme for Wireless Sensor Networks. *Sensors*, **2015**; vol. 15, no. 1, pp. 2104–2136.