Applicability of Security Measures in a Wireless Sensor Network Use Case

Martin Leuckert
University of Magdeburg
Clinic of Nephrology
martin.leuckert@ovgu.de

Peter R. Mertens
University of Magdeburg
Clinic of Nephrology
peter.mertens@med.ovgu.de

Gunter Saake
University of Magdeburg
Department of Computer Science
saake@iti.cs.uni-magdeburg.de

ABSTRACT

Wireless Sensor Networks (WSN) are used widely and can be found in a growing number of heterogeneous applications. Consequently, the amount of data produced by these systems increases tremendously. Most data produced by sensor networks is confidential and needs corresponding security measurements. There are many different approaches to maintain data confidentiality; homomorphic cryptosystems, for instance, became a popular research topic for WSN and Body Area Network (BAN) even though they were deemed too costly a few years ago. In the present paper, a selection of homomorphic encryption approaches will be compared to each other as well as to other current standard security measurements. The objective is to evaluate the data security, computational effort, memory requirements, key distribution (if necessary), energy consumption, and usability - from sensor to final storage.

General Terms

Measurement, Performance, Design, Experimentation, Security, Human Factors, Legal Aspects.

Keywords

Data Confidentiality, Homomorphic Encryption, Wireless Sensor Network, Body Area Network.

1. INTRODUCTION

Sensor Data are collected in very heterogeneous environments. Landscape observing systems or military applications of Wireless Sensor Networks (WSN) are by nearly all accounts different from sensor systems in a Smarter Home environment. However, they all produce confidential data. Particularly if the sensors are recording biometric traits or medical data in a Body Area Network (BAN), confidentiality, integrity and privacy need to be addressed. Due to the diversity of the systems, the possibilities for security measurements are not identical. Battery powered sensors of neither WSN nor BAN are deemed able to implement a highly sophisticated asynchronous cryptosystem as, for example, a modern 'IoT-device' in a medical application can. Medical data streams throughput should not be slowed down significantly by heavy encryption processing, but it probably has higher computational power and does not necessarily rely on battery power.

Until now, there have been few real applications for homomorphic encryption due to the high demand in computational effort, space requirements, and noise [12]. However, recent advances in the field of homomorphic encryption show that it is becoming a viable solution to achieve end-to-end confidentiality in wireless sensor networks [7, 11]. This paper will start a discussion on whether this is still a long way off or an applicable option.

2. USE CASE

Saxony-Anhalt has approximately 14,9% diabetics [1]. Diabetes amplifies the probability and intensity of developing nerve damage, especially in extremities. Since this is an iterative process, affected people are not aware of the nerve damages and lack bodily responses like pain. As a result, they tend to stress their feet incorrectly, which can cause the common secondary disease called 'diabetic foot syndrome'. A diabetic foot syndrome leads to inflammations (called ulcer) and oftentimes to amputations.

2.1 Smart Prevent Diabetic Feet

Armstrong et al. identified temperature increase in the respective foot regions up to five weeks before an inflammation occurs [3]. Based on these findings, the "Smart Prevent Diabetic Feet" study intends to monitor foot temperature. For this purpose, a specially manufactured insole with six temperature sensors distributed over the insole at the positions shown in Figure 1 is used. The insoles are given out to 150 test persons who will be measuring twice a day for two years. The signals sent by the insoles are then transferred to a smart device using the Bluetooth V4.1 interface. The smart device running an Android App will use the data to evaluate whether an ulceration is about to develop. For this evaluation of the data, several threshold-based algorithms are used. For instance, opposing sensor values are compared. To improve the results, abnormal influences like a heater are filtered out by including historic progression as well as performing 'sanity checks' on the data.

As this is medical data, it is very valuable and can reveal a patient's diseases like diabetes. Therefore, the sensor data produced in this setup needs to be protected. In the state of the art section the de facto standard for this kind of study as well as other security measures will be introduced.

30th GI-Workshop on Foundations of Databases (Grundlagen von Datenbanken), 22.05.2018-25.05.2018, Wuppertal, Germany.

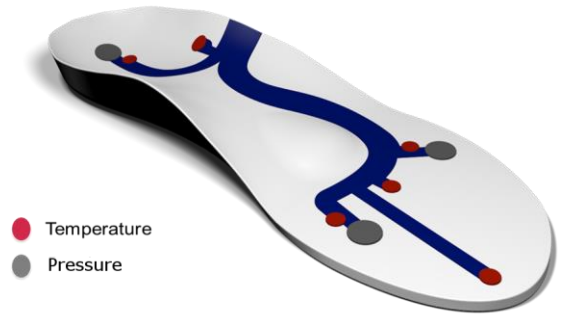

Figure 1. Sensor layout of Smart Insole.

While this example only considers foot temperature, it represents an entrance into the topic. Many more health-monitoring applications, e-Health or "Ambient Assisted Living" systems produce or work with a patient's data. However, many of them rely on an intermediate system – often called 'aggregator' or 'IoT-device' – like a smartphone or a tablet that gives meaning to the collected sensor values.

3. STATE OF THE ART

Previous research discusses both symmetric as well as asymmetric encryption. However, it was rather common to decline asymmetric encryption due to high computational effort and energy consumption [12]. Especially Clinical studies like Smart Prevent Diabetic Feet are obligated to comply with data security laws. To meet this requirement, this study is performed as a blinded study. Accordingly, all personal data are pseudonymized and transportation of the data is protected from eavesdropping or man-in-the-middle attacks by implementing the standard encryption protocol AES-128/256 for both the Bluetooth V4.1 interface and the HTTPS data transfer.

One could argue that personal data encrypted with AES-256 is secure; however, the implementation of the cryptosystem may have flaws. On the other hand, a major issue is the security on the smart device, which is vulnerable to social engineering. Therefore, to assure that the data is always protected, the data should never be available as plain data. To prevent the access of plain data, the capturing devices are required to encrypt the data, or the mobile device immediately encrypts the incoming data. This paper will propose different setups and usages of encryption approaches. This will include expectations regarding advantages, flaws and feasibility of each setup.

There is a broad variety of studies looking into node distribution as well as the path and connectivity between nodes, but this is out of the scope of this paper. Instead, the research presented in this paper is based on previous studies on encryption approaches and security investigations. One of the main issues regarding security in wireless sensor networks is the key distribution. Chen and Chao's survey of key distribution schemes in WSN [9] are a valuable introduction into the topic.

There is a variety of (lightweight) encryption algorithms that can be used in BAN to protect the confidentiality of the data. Next to the AES, which was mentioned before, there are the 'Tiny Encryption Algorithm' (TEA) as well as lightweight block ciphers like HIGHT, PRESENT, Prince and KLEIN to name a few. Each of them has different advantages and disadvantages and different areas of application [5]. However, all of them are considered suitable for the use of embedded systems with limited resources, whereas asymmetric approaches, for instance the 1024-bit RSA, are less suitable for a very small sensor device running on battery power.

4. Problem Formulation

Acquisition

Wireless Sensor Networks and Medical Applications produce tremendous amounts of confidential data. However, many of these applications open up severe security loopholes [13]. Especially on the sensors' side, data security is sometimes non-existent. Due to their differences in cost efficiency, there are not many security measures that can be applied. Apart from computational limitations, there are RAM constraints as well, which require block ciphers optimization of memory consumption. Energy consumption also remains an issue that concerns most autonomous sensors, because renewable energy is absent.

The most likely attacks are impersonation or denial-of-service attacks [2, 4]. Usually, there are no backups for a sensor in a BAN; the denial-of-service attack may prevent that vital life signals are monitored. The impersonation attack on weak encrypted or even plain signals may disclose personal information. In Wireless Sensor Networks, if a node is compromised, it usually will be excluded from the cluster in case a malicious intent can be detected. BAN do not have these possibilities since they do not have any redundancy. This means either that a compromised node leads to a data breach or even a shutdown of the compromised system.

Communication

For saving resources, communication protocols like Bluetooth Security Level 1 could be used, which do not implement any security measures at all. In the context of highly confidential data, as in the use case described above, this cannot be used. To prevent attacks like 'Man-in-the Middle' or 'Eavesdropping', Bluetooth Security Level 4 ("Authenticated LE Secure Connections pairing with encryption") is used. This level requires a connection that must be encrypted using the Secure Connection Pairing, which was introduced in Bluetooth LE version 4.2. However, this does not address already encrypted communications. If, for instance, a homomorphic encryption approach is applied, the additional effort to encrypt the data again through the Bluetooth protocol produces unnecessary overhead. Going back to Bluetooth Security Level 1 makes sense in that case; however, the authentication issue and impersonation still need to be addressed.

Storage

There are major differences in terms of storing and working with encrypted data and plain data. While the assumption used to be that a server will be compromised at some point, it is not recommendable to trust a cloud system to begin with. The confidentiality and integrity of data is at risk at all times as long as no proper security measurements have been implemented. As each system faces individual threats, a fitting security scheme requires a threat analysis. If the outcome is, for instance, that a certain encryption shall be used, the question arises what this means for

the database and the performance. Furthermore, if the data in a database stays always encrypted, how does this affect the query performance, the storage requirements, and the design of the database? The different setups described in the following chapter shed light on the impact on usability, security, and performance.

5. APPROACHES AND IDEAS

Security investigations usually compare different cryptosystems against each other and look at the code length, block size, and the number of rounds, etcetera. This paper will take the setup as a whole into account in order to identify additional parameters that might influence the data security and performance. On the one hand, different cryptographic algorithms will be used; on the other hand, the setups will encrypt and decrypt at different stages of the process. Because of the differences, each setup is expected to have different security and performance properties. Next to analyzing possible attacks on the systems, the applicability for each involved device will be examined: This starts at the sensor device that records data, continues at the aggregator, and finished at the database. This also includes factors like the utilization of resources (for instance computational effort or memory while computing), key distribution, the required time until data is available, and maintainability.

One of the setups will not use encryption and instead implement the current data security requirements, which can be fulfilled by pseudonymization of the data. This setup is expected to be the best in terms of computation times, time until availability and energy consumption. However, it is also likely that it poses the greatest risk to the data. Some setups will implement symmetric encryption algorithm AES and lightweight encryption algorithms from section 3.

For the asymmetric encryption, the additive homomorphic Paillier cryptosystem and an elliptic curve algorithm as well as the fully homomorphic encryption 'THFE' from [7] are considered. The Paillier is a good example of homomorphic encryption as it can be utilized for Similarity Verification without decrypting the data [8, 10]. However, it is expected to be too expensive compared to lightweight symmetric cryptographic algorithms. The paper and the open-source implementation in [7] raise high expectations and will therefore be compared to the performance, security and advantages and disadvantages of the Paillier cryptosystem as well as the lightweight symmetric cryptosystems.

Expected outcomes are statements to each investigated approach regarding the resilience against compromises, the resource efficiency, and further advantages and disadvantages.

Resilience against compromises

While the intention is not to guarantee that a node is not compromised, the key distribution (where applicable) must be secure at all times.

There is a variety of standard cryptanalysis that could be used to compromise the system. A few exemplary attacks are 'Brute Force', 'Ciphertext-Only Attack', and 'Man-in-the-Middle Attack'. The authors of [4] categorize the following examples as 'Modern Attacks': An attacker could exploit 'Operating System Flaws' or 'Memory Residues', 'Temporary Files', 'Differential Power Analysis', 'RSA-155 (512-bit) Factorization'. The examples target different levels and stages of a system. Some attacks target the encryption process and others try to exploit vulnerabilities of the communication. This means that the selection of a 'strong cryptosystem' will not necessarily result in a secure system.

Even if an attacker would not be able to decrypt ciphertext, just listening to when a communication is established could result in the exposure of habits, like, for instance, when a patient is at home. Vice versa: If an attacker knows when a patient is at home, they could use this information to identify which dataset can be associated with a patient.

Resource efficiency

Encryption and key management should not have a huge impact in cases of limited resources. Usually, asymmetric approaches are not even considered, because they are deemed too expensive. Furthermore, the communication is rather cost heavy, which means that fewer messages exchanged with as few packages as possible should be aimed at.

Advantages & disadvantages

Are there side effects brought along by the security measures? What operations can be performed on each stage of the process? What are the side effects: e.g. searching in encrypted space is much slower, takes more space etc.

5.1 Setup description

This section describes the considered scenarios, their pros and cons, security, and applicability.

5.1.1 Scenario 1: solution of use case

The first setup corresponds to the current solution, which means that only the data transfer is encrypted. The transfer protocols implement state-of-the-art encryption, which significantly hampers eavesdropping. However, each device can potentially be compromised in a way. The sensors may face imposter attacks, the smart device could be compromised by malware and the server hosting the database may have an ill-intending administrator. The data is still pseudonymized, but both the sensors and the IoT-device can easily be linked to a single person as they always carry it.

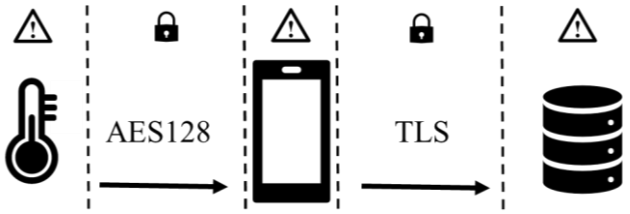

Figure 2. Scenario 1 – standard communication protocols.

5.1.2 Scenario 2: encryption on IoT-device

The imposter attacks described in scenario 1, see Figure 2, require exchanging the sensor devices, flashing the device's firmware, or getting very close with an impersonating device due to the low range of the Bluetooth signal. At least for this use case an imposter attack is rather unlikely; therefore, the following two scenarios are considered: The sensor transfers encrypted data to

the smart device, the data will be decrypted and immediately be re-encrypted using a homomorphic encryption approach.

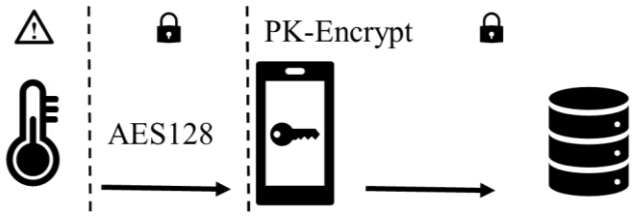

Figure 3. Scenario 2 – encryption on IoT-device.

The unencrypted data will never be stored on the device and is only available in memory for a short duration – just long enough to perform the pre-analysis of the data from Section 2. The device has more computational power and energy available than the sensor device and is expected to be able to perform the required calculations in reasonable time. The data will not be decrypted before the transfer to the database server and therefore stays secure.

5.1.3 Scenario 3: immediate encryption

In scenario 3, see Figure 4, the sensor device already encrypts the data using a public-key cryptosystem (PK). A major difference between approach 2 and 3 can be found on the server side: In approach 2, the data stays encrypted as is. This prevents most attacks if the data never gets decrypted and the key is not compromised. A disadvantage lies in the usability of the data, as it is not searchable or indexable. This is different in approach 3a / 3b, where the data will be decrypted on arrival. In 3a, the data will immediately be re-encrypted on arrival using a searchable encryption scheme. For comparison, the data will stay unencrypted in approach 3b.

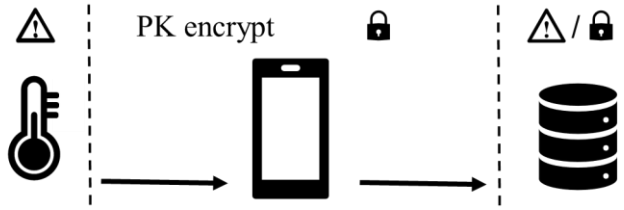

Figure 4. Scenario 3a/b – immediate encryption.

In the use case from Section 2 the IoT-device is expected to perform a pre-analysis of the data based on thresholds. This operation can only be performed on the homomorphically encrypted data. Most of the known homomorphic encryption schemes are asymmetric. For use cases that do not require the aggregator to work with the encrypted data, a symmetric approach could be used, but this is out of the scope of this paper.

As was mentioned before, the device is required to perform a pre-analysis of the data and decide whether the user's temperature data is within a certain range or not. This can be done due the homomorphic encryption approach. Some homomorphic cryptosystems can be exploited in order to implement a "Similarity Search" [8], which allows the device to decide about the user's health status without decrypting the data.

5.1.4 Scenario 4: always encrypted

A fourth scenario, see Figure 5, could be a mixture of 2 and 3; the data will be encrypted as soon as possible using a homomorphic public key cryptosystem and never decrypted. This appears to be the strongest solution in terms of resilience against compromises because the data is almost never available as plain data. On the other hand, encrypted data may still be vulnerable to key leakage, brute force attacks, timing attacks, loopholes in the cryptosystem or implementation and many other threats. Additionally, in scenario 4 all analysis of the data must be performed on the encrypted data, which is very time and resource consuming. Then again, the additional computation is performed on the backend system, which has more available resources than the IoT-device.

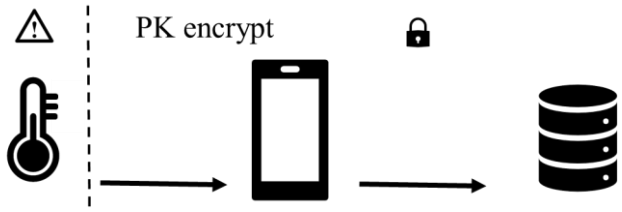

Figure 5. Scenario 4 – always encrypted.

The fourth scenario is different from Scenario 3, see Figure 4, because the received data is not searchable and not indexed. Therefore, working and analyzing with the data is hardly possible if it stays encrypted, because only limited operations like, for instance, a predefined set of threshold-based operations are possible, whereas other scenarios allow data mining processes to improve the ulcer detection algorithms from Section 2.

Assuming the corresponding private key is stored at a trusted third party instead of the backend that stores the encrypted data, this approach prevents data leakage at server-side, because even if an attacker could gain access to the data, they cannot decrypt it.

5.2 COMPARISON

This section summarizes the expected results of a security and performance analysis of the different setups. Various parameters can influence the results. For instance, increasing the key length will improve the security of a cipher but has negative side effects like reducing the possible throughput and increasing memory consumption.

Table 1. Expected performance of the scenarios (-- equals very bad / non-existent, ++ equals very good).

Scenario	Resilience against compromises	Resource efficiency (collectively)	Utility
1	--	o	++
2	o	+	+
3a	+	o	+
3b	o	+	++
4	+	-	o

Table 1 does not necessarily reflect the properties of each applied cryptosystem, but the investigated system it is used in. For

example, the cipher produced by AES256 can be assumed to be resilient against brute force attacks, but there could still be an imposter sending the cipher. The possibility of an imposter attack is a security flaw of the system rather than of the applied cryptosystem.

It is not possible to just eavesdrop the communication in approach 1 because the messages are encrypted, but, on every other component, the data is available as plain text. For this reason, it is very prone to compromises of many kinds. This is different in all other setups; however, each has its own advantages and disadvantages and no setup is expected to be perfectly secure. Furthermore, as stated in the introduction of this section, there are different requirements to different use cases. For example, a sensor network with very high throughput and near real-time availability requirements must not be delayed by too much because of computational overhead.

The databases stay unencrypted in approaches 1, 2, and 3b. Medical data protection for clinical studies require the implementation of proper access control and implementing pseudonymization and a locally secured room for the servers. When these requirements are accomplished the data is somewhat secure because it is hard to gain unauthorized access. Assuming an attacker managed to get access to the study database from the use case of Section 2, they cannot immediately tie data sets to a person because the datasets are pseudonymized. This is different if the attacker already has background information about the probands because they could use their knowledge to look for habits or patterns of the patients, e.g. when they usually go to church, to identify their pseudonym and therefore their datasets. Nevertheless, it would leak information about the study, or, in different use cases, other meta information. Here again, this is a question for risk assessment and applicable law, which can play different roles for each use case. In Table 1, this is reflected by decreasing the security score compared to encrypted databases, because it is less secure. On the other hand, the usability of plain data or searchable encryption schemes are better: Queries, indexes, transactions and stored procedures are easily processed because information about the data is given or obtainable.

Table 2 gives more in-depth information about the resource consumption per component. The first scenario only uses standard protocol exchanges, which is why it is not affected by the selection of cryptographic function. All other scenarios are affected by this.

Table 2. Resource consumption per component.

Scenario	Sensor device	Aggregator	Backend
1	Medium	Low	Low
2	Medium	High	Medium
3a	Very high	High	Medium
3b	Very high	High	Low
4	Very high	High	Medium

The values for the aggregator and backend include encryption, decryption and homomorphic operations for the pre-analysis (each where applicable); on sensor device side the encryption effort is looked at. The limiting resource factors for each component can vary depending on the performed operations, the selected encryption (function and key length) and the available resources per component. Due to this, there can be different limiting factors like, for instance, CPU, RAM or network adapters (esp. Bluetooth). Hardware accelerations are not used.

If a similar threshold-based verification like Rane et al. introduced in [8] is used, the verification takes place on a trusted third party. However, this does reduce the required effort for the aggregator or backend because there are still homomorphic operations prior to the verification itself, which have a similar effort.

The tables should be understood as a hypothesis of this work. It changes depending on requirements of use cases, availability of resources, and applied cryptosystems. Furthermore, it can be extended by additional setups and requires analyzing each component (sensor device, IoT-device, backend, and communication channels) individually.

6. CONCLUSION

All proposed setups have different properties and their suitability does not only depend on the computational power of the nodes but also the quantity of collected data, expected real-time availability, data security requirements and many more. There are vital parameters recording applications, which require special safety and security precautions, while, on the other hand, a forest fire detection application, probably is not as demanding.

The next step will be to implement the proposed setups and vary the parameters. This includes sensor signals and different encryption approaches with varying key length. An expected outcome of the investigation is a collection of statements regarding the requirements of a setup as discussed in section 1, the data security including possible attacks, as well as usability of the data (can it easily be queried, etc.).

Another important step would be to retrieve more general statements about the applicability of the proposed setups. Where else can they be applied? What if the database is not located in a trusted environment, e.g. in a cloud system?

7. ACKNOWLEDGEMENTS

This work was funded by the project „Smart Prevent Diabetic Feet“ (DRKS00013798) - EFRE (2016 - 2018): Forschungsverbund “Autonomie im Alter”.

8. REFERENCES

[1] Heidemann, C., Kuhnert, R., Born, S., Nave, C. “12-Month prevalence of known diabetes mellitus in Germany”, Journal of Health Monitoring, 2017.

[2] Al Ameen, M., Liu, J. & Kwak, K. “Security and Privacy Issues in Wireless Sensor Networks for Healthcare Applications”, Journal of Medical Systems, 2012.

[3] Armstrong, DG., Holtz-Neiderer, K., Wendel, C., Mohler, Jane, M., Kimbriel, Heather R., Lavery, Lawrence A. “Skin temperature monitoring reduces the risk for diabetic foot ulceration in high-risk patients”, The American journal of medicine, 2007.

[4] O'Hanley, R., Tiller, J. S., “Methods of Attacking and Defending Cryptosystems” Information Security Management Handbook, Sixth Edition, Volume 7, 2013.

[5] Singh, S., Sharma, P.K., Moon, S.Y. et al. “Advanced lightweight encryption algorithms for IoT devices: survey, challenges and solutions”, Journal of Ambient Intell. Human Comput, 2017.

[6] Castelluccia, C., Chan, A. C-F., Mykletun, E., Tsudik, G. “Efficient and provably secure aggregation of encrypted data in wireless sensor networks”, ACM Trans. Sen. Netw. 5, 3, Article 20, 2009.

[7] Chillotti, I., Gama, N., Georgieva, M., Izabachène, M. “Faster Fully Homomorphic Encryption: Bootstrapping in Less Than 0.1 Seconds”, Advances in Cryptology – Asiacrypt 2016, 2016.

[8] Rane, S., Sun, W., Vetro, A. “Secure similarity verification between homomorphically encrypted signals”, U.S. Patent 8249250 B2, 2012.

[9] Chi-Yuan Chen, Han-Chieh, Chao. “A survey of key distribution in wireless sensor networks”, 2011.

[10] Dargad, A., & Sutar, S. “Enhanced Data Leakage Detection in IoT Network Backup with Cloud Using Paillier Homomorphic Cryptosystem”, Asian Journal of Convergence in Technology, 2018.

[11] Brakerski, Z., Gentry, C., Vaikuntanathan V. “(Leveled) Fully Homomorphic Encryption without Bootstrapping”, ACM Trans. Comput. Theory 6, 3, Article 13., 2014.

[12] Lauter, K., Naehrig, M., Vaikuntanathan, V. “Can homomorphic encryption be practical?”, Proceedings of the 3rd ACM workshop on Cloud computing security workshop, 2011.