# Group Key Management in Wireless Sensor Networks: Introducing Context for Managing the Re-keying Process

Andreas Lyth
USN Norway
andreaslyth@hotmail.com

Radmila Juric
USN Norway
Radmila.Juric@usn.no

Daniel Larsson
USN Norway
Daniel.Larsson@usn.no

## Abstract

*This paper proposes an algorithmic solution to Group Key Management (GKM) in Wireless Sensor Networks (WSN), which could address a single point of failure in cybersecurity. The paper moves away from the traditional (de)centralized and distributed solution in GKM and focuses on GKM decision making based on a) the context in which WSN and their nodes communicate, and b) the semantic which describes the environment where WSN and their nodes reside. The proposed algorithm defines which node, within the WSN, could start a re-keying process by generating a group key, and why/how this decision on the re-keying has been made. The algorithm is computable and thus it would be feasible to implement it in software applications built upon a set of WSN nodes in constantly changeable and dynamic mobile computing environments.*

## 1. Introduction

In the last few decades the advances in the Group Key Management (GKM) for wireless Sensor Networks (WSN) focused on (de)centralized, and distributed management of encryption/ decryption keys. However, it is also known that there is no universal solution to the cybersecurity problem in WSN, in either literature or practice. Numerous publications highlight the level of complexity of the problems associated with the cybersecurity in WSN [1,2,3] and outline that it is rather difficult to find a proof that a single point of failure in WSN has been eliminated [4].

This paper focuses on the issue of addressing the problem of GKM for WSN with the goal is to bring novelty to the field by considering:

> *"what should an algorithm, responsible for GKM, contain in terms of decision making, i.e. how, when and why we generate cryptographic keys for messages passed in WSN which may address a singe point of failure*".

The idea promoted in this paper is based on the *context*, the term coined by computer scientists two decades ago and associated with mobile and pervasive computing plus self-tuning software applications [5,6]. At that time software applications started "reacting" to the environment, i.e. *context* in which they resided. This proved to be essential if we wanted address constant changes in dynamic environments, where we started performing computing. Decision making, based on the *context*, where software applications reside is the essence of *contextualisation*, which has been performed in the last two decades in many problem domains in computer science, with proven results [7,8,9]. We can mirror the same situation in WSN environments. They are highly dynamic and create numerous (probably unpredictable) *contexts* in which they operate.

It is hard to believe that constant changes in WSN do not affect all existing solutions, which claim to guard GKM and address the single point of failure in WSN. The vast literature, which originally started with centralized and decentralized GKM, but now focusing on block-chain and clustering, in order to address the problem, do not bring anything new to helping with GKM and WSN vulnerability. They are mostly very old and tried software technologies and ideas, suitable for the 80s and 90s and used in software engineering, far before wireless and mobile technologies matured. Using clustering in particular in GKM is not prudent and should not be encouraged. Clustering is a recycled old idea from computer science during the 60s, when we used powerful mainframes in order to start addressing storage and operating system functions on mainframes, data management and potential distribution of computer programs, far before database management systems were invented. By applying the same ideas in a completely different environment: highly dynamic and probably un-predicable WSN of the second decade of the 21st century, we are actually using the ideas, which are not fit for purpose.

We would argue that there is no way forward in resolving problems in cybersecurity by recycling old technologies which do not address the most precious nature of our modern computer networks: dynamics and constant, un-predictable changes. In our connected world of Internet of Everything, where we join on an ad-hoc basis, and compute on devices which can be found at an unexpected "places" (e.g. from cyborgs to coastal defense drones), how could we talk about cyber security if we do not incorporate the semantics of such environments into cyber-security algorithms? If we

URI: https://hdl.handle.net/10125/80234
978-0-9981331-5-7

expect that computer science would answer to the modern problems of GKM in WSN, then we should interweave the semantic of this constantly changing environment into computational models, which use those semantics. Consequently, contextualization of decision making in GKM may be one of the ways forward and the abandonment of the old, unsuccessful ideas might not be a bad thing to do.

At the time of writing this paper, no published sources have been found, which define and manipulate context where cryptographic keys are created and managed, as a part of GKM practices. Therefore we are not able to juxtapose or challenge our ideas with any other work. However, *contextualization*, as defined in computer science, without any doubt is one of many possibilities, which could be used by academics/ practitioners in order to move forward in the complicated world of cybersecurity.

What would be the contribution of the contextualized decision making in this problem domain, as advocated in this paper?

First, the computational algorithm for GKM, must be based on the *context*, i.e. the semantic collected at a particular moment, as a part of the GKM, which secures that decision making on the re-keying process is dependent on that *context*. This may be a situation extracted from WSN, in terms of WSN nodes, families and current, valid and potentially shared keys. However, the context is an open term, which illustrates all possible situations we can experience within WSN.

Second, the *context* must be defined and formalized in advance and, as such, should encompass relevant semantic for managing GK and secure computations of the management through formalism. In this paper, we introduce Family Key Paradigm and End-time ticket, as a *context*, within which we operate and perform algorithmic decision making in GKM. This is by no means the only *context* in which one can exercise contextualization. However, this one gives the semantic of the GKM environment through its paradigm and uses numerical values (as opposed to textual information) for defining/manipulating end-time ticket.

Third, the algorithm which would support decision making in GKM must not be obscured, it must be explainable and computable. Therefore, decision making must be based on simplicity (and not complexity) of examining numerous conditions within a WSN, delivered through the semantic stored in the paradigm, and making decisions based on evaluation of these conditions. "True" evaluations could take us to safe decision making in GKM, but "false" verdicts require attention. Therefore the robustness of the algorithm is in managing "false" verdicts, which in turn would depend on the actual "context" in which an algorithm runs (the status of the Family Key Paradigm and the values in End-time ticket). "False" verdicts are always a signal that we either have to react, include human intervention, or change the decision on the GKM, but they must not be obscured in any formalization and in any algorithm.

Finally, there is one simple aspect of GKM which should be achieved. It should be difficult, if not impossible to guess, or to know in advance, when, why and how a new key is generated within a family of nodes, or across a set of families in WSN. This decision would always depend on the exact *context*, and this is when we can utilize it to its maximum: the detected change(s) in the WSN, change the context and in turn would require changes in the GKM by, for example, triggering the start of the re-keying process. It would be impossible to predict or guess, when and where the re-keying process starts, without understanding the relevant *context,* which triggers the change. If we can at least achieve this, we will be able to look at the issue of single point of failure completely differently. Most of our current worries such as trusting nodes, compromised nodes, key confidentiality and similar become immaterial.

The paper is organized as follows. Section 2 looks at the complexity of this research problem, summarizes its challenges and defines the main research question. Section 3 formalizes the proposal by giving definitions of Family Key Paradigm and End-time ticket, proposes and illustrates the algorithm and outlines the outcome from the proposed contextualization. Section 4 overviews the inadequacy of using block-chain and clustering mechanism in modern WSN and GKM, section 4 evaluates the research and section 6 defines future work.

## 2. The Research Problem

The complexity of the cybersecurity problem, cryptography and GKM is well known. It is particularly challenging in WSN and constantly changeable environments, such as the Internet of Things (IoT), where devices and WSN nodes cohabit, exchange messages and utilize the infrastructures of WSN [10,11].

Having strict control of the process of re-keying in GKM, as is in (de)centralised schemes might seem reassuring in the management of cybersecurity. However, centralisation in computer science does not necessarily create flexible solutions [12,13] and thus it might be seen as a rigid way of protecting our data and traffic across WSN. Also, centralization may impose security risks, as monitoring of data traffic through eavesdropping may reveal critical information necessary for carrying out an active attack, such as DoS-ing, on the node responsible for re-keying in GKM [14]. In the worst-case scenario, disabling the affected node

might provide the attacker the additional time needed to break the cryptographic key. Distributed and collaborative schemes mitigate this threat, but they are afflicted with the same strictness in the process of re-keying in GKM. In collaborative schemes, we know that all the nodes in a group will contribute to the re-keying process, and the distributed scheme, as defined in [15], is affected by the same weakness because an attacker will know that the last joining node will be responsible for re-keying.

Cryptographic keys are of a finite space, and everything you can hope to "keep as a secret" is the cryptographic key. However, all cryptographic algorithms, except "one time pad" are possible to break [16,17]. Due to space restriction we do not elaborate on the cryptographic algorithm metrics as in [18]. Also, the threat of quantum computing, combined with Shor's algorithm, posed on modern cryptographic algorithms [19] with respect to Rivest-Shamir-Adleman encryption, is real and thus relying on cryptographic keys in resolving cybersecurity problems has to be completely revisited.

Furthermore, the challenges of GKM are also numerous and include

(i) The re-keying, as a measure to maintain forward and backward secrecy, and
(ii) Scalability [11], which occurs when managing an increasing number of nodes in the WSN [16].

The problem of single point of failure is another challenge in any GKM-scheme where a single entity is responsible for managing a network, as in in centralised schemes from [4]. Any issue leading to a failure in the managing nodes/keys might lead to grave security issues such as failure of re-keying or managing nodes in a network. The problem of single point of failure for decentralized solutions [20] may allow a single node to be responsible for group key management. Some distributed schemes contribute in generating a cryptographic group key, thus avoiding the problem of single point of failure by design.

Other problems in distributed GKM, have been explained in [14]. They argue that nodes in distributed schemes must be aware of all the other group members in order to be able collaborate or distribute the group key and thus requires more memory in each node. However, distributing the workload among the nodes, leads to the higher processing time and communication costs [21], as the number of rounds and exchange messages between members during group operation grows.

In summary it is reasonable to address these research problems by focusing primarily on mitigating a single point of failure, when managing the re-keying process in GKM. It is also important to rethink both:

a) deciding when the re-keying process is going to occur, and

b) selecting the network node group member(s) which would be responsible for carrying out the re-keying process.

Therefore the problem of cybersecurity in WSN could be scaled down to the problem of GKM, which would eventually eliminate a single point of failure.

## 3. The Proposal

The paper proposes a shift of thinking about the re-keying process within GKM, where the *context*, i.e. the semantic of the environment in which WSN exist would determine how/when the re-keying process may start and which algorithm it will follow.

There are four different aspects in this proposal. First, it exploits the semantic of the context, in a particular moment or in a specific situation, which would determine where (in which node) and when (in time) the cryptographic group key will be generated. Second, the algorithm for the GKM, and re-keying in particular, should be based on the premise that each node in a WSN is capable of generating a random number to be used as parameter for a mathematical function to determine the maximum time a cryptographic key can be in use. This shall correspond to a time frame, according to the semantic defined in the same *context*. Third, we could consider that the existence and role of a network provisioner [17] may play an important role in defining the *context*. We should explore its role: from its traditional place in (de)centralised GKM to the potential reasoning mechanism, which would define the exact context for deciding about re-keying in GKM [9]. Fourth, the issue of "randomness" is an important principle in cryptography. By adding levels of randomness in GKM we can obscure the attack surface of the WSN and make the work of the cryptanalyst harder.

The proposal consists of three parts.

- Definitions of WSN nodes and their families as a path to establishing the context in which WSN operates.
- Definition of the *context*, named as Family Key Paradigm with end-time-tickets for each node, which in turn determines the length of the validity of any key
- The algorithm itself which manages the re-keying process.

### 3.1. Definitions of Family of Nodes

Let us assume that we have up to $n \in \mathbb{N}$ uniquely and addressable, and authorized nodes $\{N_i \mid i = 1, 2, 3, \dots, n\}$ in a network which are grouped by a

provisioner P to up to $m \in \mathbb{N}$ families $\{F_j \mid j = 1, 2, 3, \ldots, m\}$.

For simplicity reasons and due to space restrictions we illustrate this definition by having 5 nodes per family. There is no particular scientific rationale behind using 5 nodes, but this appears to be a minimum for illustrating the ideas from the proposal and easy to draw. Neither context definition nor algorithm depends on exact number of nodes.in a WSN.

Each node within a family is one hop reach of any other node in that family. Figure 1 a) shows that all nodes in one family is within one hop reach from each other. Each family $F_j$ must have a symmetric cryptographic key $K_j$, for encrypting messages $E_{K_j}(m_j) = c_j$ , and decrypting messages $D_{K_j}(c_j) = m_j$, sent to a node in the family. Before encryption the address of the intended recipient node must be added to $m_j$ This is illustrated in Figure 1 b).

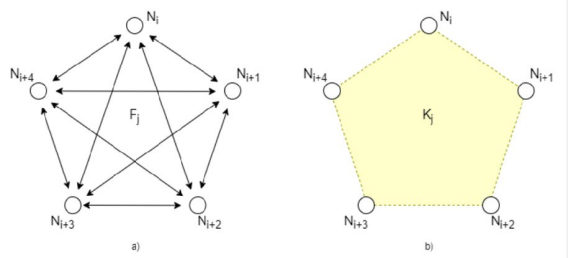


*Figure 1: Illustrations of a family of nodes.*

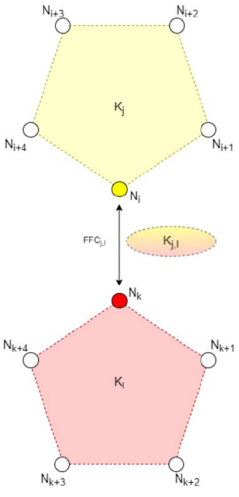


*Figure 2: Illustration of family friendly connection, between two families.*

A node $N_i^{(j)}$ in family $F_j$ may establish one, and only one connection to one other node $N_k$ in a different family $F_l$. The two nodes must be within a one hop range of each other. The connection between $N_i^{(j)}$ and $N_i^{(l)}$ and $N_k$, in the families $F_j$ and $F_l$, is called a family friendly connection $FFC_{j,l}$. The nodes making up a FFC connection must share a unique symmetric key $K_{(j,l)}$ for encryption and decryption of messages sent between$N_i^{(j)}$ and $N_i^{(l)}$, and consequently between the two families $F_j$ and $F_l$. Nodes that contribute in FFC's, adopt the role of intermediaries. This connection is illustrated in Fig. 2. By allowing family friendly connections, we are actually allowing extensions of the number of nodes in families: FFC may trigger a single group key for friendly families.

### 3.2. Introducing the Context: Family Key Paradigm and End-time ticket

A family key $K_j$ has a predefined time to live, defined by the time frame $t^{(Fj)}$. Correspondingly, $t^{(Fj)}$ has an upper and lower limit $t_{min}^{(Fj)}$ and $t_{max}^{(Fj)}$ defining a window within which $K_j$ must be replaced by a newly generated $K_j'$. The responsibility of re-keying the family key $K_j$ must not be fixed to one specific node. Rather, all nodes $N_i^{(j)}$ in family $F_j$ arbitrarily take turns in generating and distributing the new family key $K_j'$ among the family nodes. Ensuring this is done by implementing the end-time-ticket $t_{Nj''}$ applicable to any node.

An algorithm taking advantage of the family-key-paradigm is continuously running in all nodes $t_{Nj''}$ in family Fj . An end-time-ticket $t_{Nj''}$ is generated in the node, with a value between $t_{min}^{(Fj)}$ and $t_{max}^{(Fj)}$, as shown in Figure 3. As this happens in each node in the family at the same time, a parallel process is created wherein the first node to reach its end-time-ticket will be responsible for generating and distributing the next family key ($K_j'$ ). This parallel process is illustrated in Figure 3.

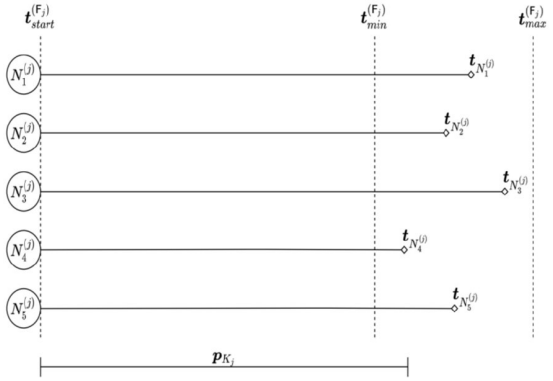


Figure 3: Family Key Paradigm with End-time Tickets

When a newly generated family key $K_j'$ is acknowledged by all nodes it starts functioning as $K_j$ and is to be used for encryption and decryption of messages sent within the family. This acknowledgement also starts a new time frame, and updates a Family-Key-Paradigm, for the family key, as described above. When a new time frame $t^{(Fj)}$ starts, each node $N_i^{(j)}$ in family $F_j$ must generate an end-time-ticket $t_{N_i^{(j)}}$ for that specific node. The node's end-time-ticket $t_{N_i^{(j)}}$ is based on a random number generated by the node. In turn, this random number is used as a parameter in a mathematical function, creating a number within the window of the time frame, defined by $t_{min}^{(Fj)}$ and $t_{max}^{(Fj)}$. It goes without saying that the generated end-time-ticket $t_{N_i^{(j)}}$ is to be kept secret by all nodes $N_i^{(j)}$ . We assume that nodes would not publicize their end-time-tickets.

### 3.3. The Proposed Algorithm

The basic algorithm, shown in Figure 4 introduces the main philosophy of the proposal. According to it, a re-keying process starts only in cases when either:

a) $N_i^j$ is informed that another node $N_k^j$ reached its end-time-ticket $t_{N_k^j}$ *or*

b) $N_i^j$ is NOT informed that another node $N_k^j$ reached its end-time-ticket $t_{N_k^j}$ BUT $N_i^j$ reaches its own end-time-ticket $t_{N_i^j}$

This means that the algorithm will run without any need to start the re-keying process until either a) OR b) becomes true.

Therefore, the re-keying process will depend on a context or situation collected through/explained by the Family-Key-Paradigm. The paradigm contains the exact information which will allow the algorithm to evaluate a) and b) above.

There are three important aspects of the proposed algorithm in Figure 4:

- It is a simple algorithm and easily explainable;
- It follows structured-programing If-THEN-ELSE sentences, which are directly computable in any language (apart from functional, mark-up and specific languages, which do not rely on control structures) and this the implementation is trivial.
- a) and b) above are implemented as main backbone / structure of the algorithm.

However, this basic idea on, how to manage GKM, given in Figure 4 should be extended with explanations on how to implement the logic which is commented as */* Explanations ....”* using blue and green and italic font. We may have various options here in order to define when the rekeying process must start, but the structure of the algorithm may never change.

**IF** ( $N_i^j$ is informed that a node $N_k^j$ reached its end-time-ticket $t_{N_k^j}$ ) **THEN DO {**

***/* Explanations on how the nodes should respond when notified that another node has reached its end-time-ticket, for re-keying the family key ($K_j \leftarrow K_j'$), and establishing a new family-key paradigm ($p_{K_j} \leftarrow p_{K_j^*}$) */***

**}**

**ELSE** ($N_i^j$ is not informed that another node $N_k^j$ reached its end-time-ticket $t_{N_k^j}$ ) **DO {**

**IF** ( $N_i^j$ reaches its own end-time-ticket $t_{N_i^j}$) **THEN DO {**

***/* Explanations on how the nodes node should react when reaching its end-time, for re-keying the family key ($K_j \leftarrow K_j'$), and establish a new family-key paradigm ($p_{K_j} \leftarrow p_{K_j^*}$) */***

**}**

**ELSE** ( $N_i^j$ did NOT reach its own end-time-ticket $t_{N_i^j}$ ) continue under current paradigm (do not re-key)

**}**

Figure 4: The Proposed Algorithm

Due to space restrictions we show only one part of the explanation commented as /* *Explanation*…. from Figure 4. The expansion of the proposed algorithm for cases when ( $N_i^j$ is informed that a node $N_k^j$ reached its end-time-ticket $t_{N_k^j}$ is given in Figure 5. In other words, Figure 5 gives an extension to the algorithm form Figure 4, which explains exactly what would happen in the “THEN close of the IF statement from Figure 4”. Consequently, Figure 5 defines decision making relevant to the re-keying process, but only in the situation when $N_i^j$ is informed that another node $N_k^j$ reached its end time ticket $t_{N_k^j}$.

The re-keying process is relatively simple: All nodes replace the current family key $K_j$ sent by node $N_i$ and all nodes generate a new individual end-time-ticket (Family Key Paradigm has been updated!).

The re-keying process, from Figure 5 would depend on a set of conditions, which need to be satisfied before the re-keying starts. These conditions are evaluated in five lines from Figure 5 which define IF sentences, and which clearly specify the conditions. Only when these conditions are all met, may the re-keying process start. Obviously, by evaluating all these five conditions, we could issue ALARMs, because the conditions may not be met and the re-keying could not start (as expected). We call them ALARMs but they could also be warnings, because they may have numerous purposes. This would depend on the situation in which we experience anomalies in satisfying conditions for re-keying.

$N_i^j$ attempts to inform all nodes it reached its end-time-ticket $t_{N_i^j}$
**IF(** $N_i^j$ informs nodes it reached its end-time-ticket $t_{N_i^j}$ )
**DO {**
$N_i^j$ attempts to generate a new family key $K_j'$
**IF(** $N_i^j$ manages to generate new family key $K_j'$ ) **DO {**
$N_i^j$ attempts to distribute $K_j'$ to all other nodes in $F_j$
**IF** ($N_i^j$ manages to distribute $K_j'$ to all nodes in $F_j$) **DO {**
All nodes attempt to acknowledge to $N_i^j$ they received $K_j'$
**IF** (All other nodes manage to acknowledge to $N_i^j$ that they received $K_j'$) **DO {**
$N_i^j$ attempts to instruct all other nodes in $F_j$ to abolish current paradigm ($p_{K_j}$)
**IF** ($N_i^j$ manages to instruct all other nodes in $F_j$ to abolish current paradigm $p_{K_j}$) **DO** {
All nodes acknowledge to $N_i^j$ they abolished current key paradigm $p_{K_j}$
**IF** (All nodes manage to acknowledge to $N_i^j$ they abolished current key paradigm $p_{K_j}$) **DO** {
All nodes replace the current family-key with the new key /*($K_j \leftarrow K_j'$)*/
All nodes generate a new individual end-time-ticket /*$p_{K_j} \leftarrow p_{K_j}^*$*/ }
**ELSE** (All nodes did not manage to acknowledge to $N_i^j$ they abolished current key paradigm $p_{K_j}$)
Issue Alarm 6: Abolishment of Current paradigm $p_{K_j}$ was not acknowledged by all nodes }
**ELSE** ($N_i^j$ did not manage to instruct all other nodes in $F_j$ to abolish current paradigm $p_{K_j}$)
Issue Alarm 5: Abolishment of Current paradigm $p_{K_j}$ was not instructed }
**ELSE** (All other nodes did not acknowledge to $N_i^j$ that they received $K_j'$)
Issue Alarm 4: Distributed family key $K_j'$ not received by all nodes }
**ELSE** ($N_i^j$ did not manage to distribute $K_j'$ to all other nodes in $F_j$ )
Issue Alarm 3: Newly generated family key $K_j'$ was not distributed }
**ELSE** ($N_i^j$ did not manage to generate new family key $K_j'$ **)**
Issue Alarm 2: New family key $K_j'$ not generated }
**ELSE** ($N_i^j$ did not manage to inform all other nodes it has reached its end-time-ticket $t_{N_i^j}$ )
Issue Alarm 1: Re-keying process was not initiated }

Figure 5. Algorithm for Re-keying: $N_i^j$ Reached its End-Time-Ticket $t_{N_i^j}$

### 3.4. Illustration of the Algorithm

**IF(** $N_i^j$ is informed node $N_k^j$ reached its ETT $t_{N_k^j}$)**DO** {
$N_k^j$ generates a new family key ($K_j'$)
$N_k^j$ distributes $K_j'$ to all other nodes in $F_j$
**IF** (All nodes acknowledged to $N_k^j$ they received $K_j'$) **DO {**
$N_k^j$ instructs nodes in $F_j$ to abolish paradigm $p_{K_j}$
**IF** (All nodes ackn. to $N_k^j$ they abolished par $p_{K_j}$) **DO {**
All nodes replace the current with new key /*$K_j \leftarrow K_j'$*/
All nodes generate their own new ETT /*$p_{K_j} \leftarrow p_{K_j}^*$*/ }
**ELSE** (Not all nodes ackn. to $N_k^j$ they abolished par. $p_{K_j}$)
Issue Alarm 7: Abolishment of Current paradigm
**ELSE** (Not all nodes acknowledge to $N_k^j$ that they received $K_j'$)
Issue Alarm 8: Fam. key ($K_j'$) not received by all }
**ELSE** ($N_i^j$ NOT inform. $N_k^j$ it reached ETT $t_{N_k^j}$) **DO** {
**IF** ( $N_i^j$ reaches its own ETT $t_{N_i^j}$) **DO {**
$N_i^j$ informs all nodes in $F_j$ it has reached its ETT $t_{N_i^j}$
$N_i^j$ generates a new family key ($K_j'$)
$N_i^j$ distributes $K_j'$ to all other nodes in $F_j$
**IF** (All nodes acknowledged to $N_i^j$ that they received $K_j'$ ) **DO {**
$N_i^j$ instructs all nodes in $F_j$ to abolish paradigm ($p_{K_j}$)
**IF** (All nodes ackn. to $N_i^j$ they abolished par. $p_{K_j}$)
**DO** {
$N_i^j$ replace the current with new key /*$K_j \leftarrow K_j'$*/
$N_i^j$ generate a new individual ETT /*$p_{K_j} \leftarrow p_{K_j}^*$*/}
**ELSE** (Nodes NOT ackn. to $N_i^j$ abolish. par. $p_{K_j}$)
Issue Alarm 7: Abolishment of Current par. $p_{K_j}$}
**ELSE** (All nodes did not acknowledge to $N_i^j$ they received $K_j'$)
Issue Alarm 8: Distributed family key ($K_j'$) not received by all}
**ELSE** ( $N_i^j$ did NOT reach its own ETT $t_{N_i^j}$ )
continue under current paradigm (Do not re-key)
}

Figure 6: Illustrating the Proposal: An Example when $N_i^j$ is Informed that Node $N_k^j$ Reached its End-time-ticket (ETT) $t_{N_k^j}$

The algorithm shown in Figure 6 is an illustration of the proposal and shows a situation in which $N_i^j$ is informed

that another node $N_k^j$ reached its end-time-ticket $t_{N_k^j}$. *Thus* $N_k^j$ generates a new family key ($K_j'$), $N_k^j$ distributes $K_j'$ to all other nodes in $F_j$, all nodes acknowledge receiving $K_j'$, and all other nodes in $F_j$ acknowledge to $N_k^j$ that they abolished paradigm $p_{K_j}$ There are only two alarms issued in such cases: not all nodes abolished $p_{K_j}$ and not all nodes received $K_j'$.

The illustration in Figure 6 shows different possibilities when deciding which node will oversee the re-keying process and when. If everything goes "according to plan", i.e. there are no changes in the Family Key Paradigm, and all the conditions for deciding which node will generate a key were met, the algorithm will execute the re-keying process according to the semantic stored in the Family Key Paradigm. If for any reason, these conditions are not met any more, we have a range of choices: from completely disabling the automation of the algorithm and placing human intervention in charge of the re-keying process to a forceful start of a new Family Key Paradigm. These two diverse options could be very efficient and used interchangeably.

### 3.5. Outcomes of the Proposal

There are three unexpected outcomes from the proposed algorithm:

(A) The algorithm from Figure 4 is the essence of the proposal and it is NOT negotiable. However, all other extensions from the algorithm are negotiable because they need to be debatable across the cybersecurity communities to measure if we really want such a detailed examination of each possible situation, which may occur in WSN and affect the GKM. Thus the contents of Figures 5 and 6 should be debated.

(B) The proposal disclosed the severity of this problem in general, and outlines how many ALARMs can be issued. They should all attract the attention of cybersecurity specialists, because they may affect the current cyber security philosophies, which claim to resolve the same problem using different ideas (methods?). Where and how are these ALARMS described and analyzed in any other approach to GKM?

(C) Figure 5 shows the severity of checking upon various conditions, before the re-keying process may start, but this needs further attention. Should we decide in future which one of these conditions need urgent human intervention, and which we can afford to leave unattended? Should we also resolve this by extending the context in which these decisions are being made? The illustration from Figure 6 shows how easily computable (and thus implementable) the proposal is, but do we need such precision in evaluating conditions in the defined context (Family Key Paradigm)? Would the answer for this question come from a cyber security specialist or computer scientist?

In the light of the above, it would be almost impossible for an attacker, to guess in advance which node will be responsible for generating the next cryptographic family key, and also when the re-keying process will start. This will all depend on the semantic stored in the Family Key Paradigm and the values of end-tickets (which should not be broadcast!). This appears to be an ideal context in this model of GKM.

Finally Figure 6 gives an illustration of how the algorithm would work in a particular situation, which was tested during this research. It is likely that in most of these cases, in real life, we will be able easily to manage GKM using the illustrative example. For readers interested in the way ALARMs are being classified further and which role they may have in changing the semantic of the re-keying process, we refer them to [22]. However, there is still work to be done. We must debate about

(a) which level of human intervention we might need in resolving this problem and how this would affect the classification of ALARMS and

(b) to which extent we should automate decision making in cyber-security? Unfortunately, there is no consensus in the cybersecurity discipline on that.

## 4. Block-chain and Clustering in WSN

As mentioned in the introduction, no published papers focused on exploiting the semantic of the environment through contextualization in order to secure messaging by GKM in WSN. However, there was something interesting, which surfaced in exploring the literature. First, due to the popularity of block-chain technology in modern times, there are publications, which apply it to WSN and the problem of securing safe encryption/decryption. Examples are [23,24] However attractive the idea of using block-chains to distribute keys is, it requires scrutiny. WSN are highly dynamic and in many instances, such as IoT, they require a high level of flexibility, at least when creating families of WSN nodes. On the other hand, Block-chains, by their inherent nature, may inhibit this flexibility and convert our families of nodes into a set of static structures. This would not accommodate the mobile nature of applications of WSN and particularly in IoT. More work has to be done before we can assume that block-chain technology is suitable for modern mobile and wireless operating environments of the 21st century. The advances in technologies and software technologies in particular are galloping forward too fast and we should never go back in the 21st century to rather static structures in creating software solutions.

Second, there are papers which push forward a very old and tried world of clustering in computer science and use it for GKM in cybersecurity [25,26]. These ideas did appear in the past in various heterogeneous software applications, which tend to federate (as in federated databases in the 90s) and cluster computing according to software technologies or data used at that time. These ideas might come close to ideas of contextualization, but there are still huge differences.

Could clusters address the enormous dynamism of WSN and mobile computing? When creating clusters, the GKM becomes complex and thus understanding the context which explains WHY a particular cluster has been created and having a mechanism of using its elements "on the go", are essential. However, applications such as IoT can only handle lightweight computations to manage GKM. We would need a new paradigm, which could associate clusters of WSN with dynamics of pervasive computing with reasoning [9]. Even then, the question remains unanswered: Is clustering changeable? Can it model constant changes in WSN? Could it help a family of nodes defined in this formal proposal for GKM? Could the families of nodes become clusters: remove/add nodes to the family on the go, without prior notice?

In this section we overview just a few papers which hint the use of clusters. In [26] distributed GKM is used, but the authors differentiate between transport-and application-level encryption, because they also talk about confidentiality. Their clustered and distributed key management framework is a step too close to the block-chains, which means that their interesting idea of notification propagation among clusters is buried in the rigid structure of these clusters.

In [25] the authors focuses on secure cluster-based hybrid hierarchical group key agreement for large wireless and ad-hoc networks and propose a cluster-based hybrid hierarchical-group key agreement (CHH-GKA) framework based on splitting a large group into a certain number of clusters. The last member of each of the clusters is designated as a cluster head and the last member of the group is designated as the group controller. In [27] the authors propose a cognitive key management technique (CKMT) in a cluster-based mobile environment. They claim that it reduces the re-keying process, which is required for a mobile node when it enters a new location area, thereby reducing the computational overhead and enhances the scalability to large size network. Finally the authors of [28] introduce an interesting concept of cognitive networks, which may come closer to the ideas from this research and contextualization. They look at the potential for teamwork where wireless communication progresses from an individual, device-centric approach toward group and team behavior.

## 5. The Evaluation

The summary of the Proposal states that the basic algorithm from Figure 4 contains the main philosophy of GKM. The re-keying process may not start if (i) $N_k^i$ is NOT informed that another node $N_k^j$ has reached its end-time-ticket $t_{N_k^j}$ and (ii) $N_k^j$ has NOT reached its end-time ticket. Therefore this idea, together with the formal definitions of the families of nodes, and their Family-Key-Paradigm (context!) have secured enough semantic to cover the re-keying process (when it is allowed to happen). This re-keying process, from Figure 5 can take various debatable and negotiable forms. It makes decisions on which node will be in charge of generating the new family key and when. However, the algorithm reveals various situations in which the re-keying process must be stopped (ALARMS?) for many reasons:

(a) we work in cybersecurity and all anomalies must be detected, even if they are not considered dangerous,

(b) these situations might trigger user intervention, which could be welcome in cybersecurity [29]. It would challenge perceptions that automated computations can resolve problems associated with these anomalies;

(c) ALARMs are exit points from running the algorithm, which is needed to leave no stone unturned when implementing computations. It also contribute towards the completeness of the algorithm,

(d) ALARMS open doors to discussions and further evaluation of the proposal, but at this moment alarms depend on the *context!*

The next outcome from this proposal would be a discussion on how we can implement these algorithms. They are presented in a computable format. Considering that the focus in the algorithm is on decision making (and not on encryption) when managing keys, then this software application generated from the algorithms would be easily deployable within either Android/iOS operating environments, server-cloud computing, or even on cloud edges. A software architectural model could be generated from the formal conceptual model of the proposal as in [29,30,8], potentially opening doors for using many different software technologies. One would be to replace the traditional role of the provisioner in WSN with the reasoning mechanism available through Semantic Web Technologies and thus enable, for example, reasoning upon how to create

- a family of nodes according to the definitions from the proposal and
- define and maintain the Family Key Paradigm when circumstances change.

We can use computational reasoning to make decisions on the nature, severity and impact of alarms. They should result in either creating a new family-key-

paradigm or closing down the network, as two extreme but efficient actions in WSN management. In principle, managing the context relevant for the re-keying process, as in this proposal, comes very close to the management of pervasive computational spaces: which proved to be very efficient in the context management [8].

## 6. Future Work

There are numerous pathways from this research.

Firstly computer scientists should sit around a table and have discussions with cybersecurity and WSN communication specialists in order to address the GKM problem, by using software technologies which have not found its space in the cyber security field. This is particularly true for pervasive and semantic software technologies [31,8] which could have helped in context management, if there was any interest from cybersecurity discipline to use them.

Second, there should be some investment in this research, which would finance a full-scale implementation of the algorithm. It should seek the support of semantic computing as implemented in [9], and thus might address the novel trends of having cognitive WSN and their applications. It is obvious that the proposal is easily implementable in many programming environments.

Third, we should re-examine the re-keying process from Figure 5 and its illustration in Figure 6. by investigating its level of precision. This would apply to cases when we change our mind and decide to look at different contexts, i.e. move away from the Family Key Paradigm and end-time-ticket. It is important to note that we run all these algorithms in one particular moment, and the Family Key Paradigm can unexpectedly change at any moment. However, the proposed algorithm will give results according to this change, because it can not use anything else apart from the available *context,* in that moment.

The immediate work should really be on the following.

Expanding contextual decision making, by lifting responsibility away from the provisioner and removing it eventually, would be an important step forward. Having a provisioner is one of the weakest points in any GKM and it should not be maintained. Including computational reasoning as elaborated in [31] to maintain the semantic in Family Key Parading and the values of end-time tickets is feasible. However, this is not the only reason why we should use computational reasoning, as in semantic web, in order to manage the semantic of our context. Any issue with

- the number of nodes in a current family,
- potential FFC,
- initialization of GKM,
- manipulation of the history of end-ticket per each nodes, or
- interpreting the nature of interruptions (from node failing to acknowledge that it received a new key to abolishing the current key paradigm)

could all be addressed through reasoning, which will not require extra computational power. This is a very important step forward if we wish any contextualization to succeed in cybersecurity.

There are many other smaller problems which must be addressed in future. Apart from re-keying, we need to investigate if contextualization may select which nodes are to form families (groups), and how we would authenticate them. Decision making, on which nodes from families of nodes may connect through FFCs also needs attention. This is because some of these nodes are very likely to form a natural border between families. We did mention earlier that families can share the family key and thus the key either becomes a part of one family or the other (or both). However, giving freedom by expanding and changing families freely, affects the semantic of the Family Key Paradigm and, for example, it might in future increase the number of conditions to be evaluated in Figure 5. There is always a trade-off in computer science and we should keep this in mind.

This paper is written by computer scientists to cybersecurity specialists for the purpose of opening a dialog on the current shortcomings of using software technologies when addressing problems in GKM and WSN. There is no evidence there is a shift in thinking, on the horizon, in the cyber security field, regarding the management of WSN. If this paper eventually triggers a dialog, this would be the first step forward towards full exploitations of the ideas from this research. Computer scientists can see the feasibility of implementing context aware GKM. Family Key Paradigm is only one of the ideas for illustrating contextualization. We hope that cybersecurity specialist can see the opportunity to use it. Without them on board, this research will never materialize.

## 7. References


[1] W. Diffie and M. Hellman, "New directions in cryptography". IEEE Transactions on Information Theory, 22(6), 1976, pp. 644–654.

[2] C.Y. Chen and H. C. Chao, "A survey of Key Distribution in Wireless Sensor Networks", Security and Communication Networks, 7(12), 2014, pp. 2495–2508

[3] A. Sgora, D. Vergados, and P. Chatzimisios. "A Survey on Security and Privacy Issues in Wireless Mesh Networks", Security and Communication, 9, 2016; pp 1877-1889.

[4] A. Piccoli, M. O. Pahl, and L. Wüstrich, "Group Key Management in Constrained IoT Settings". In 2020 IEEE Symposium on Computers and Communications (ISCC), 2020, p.p. 1–6.

[5] K. Barrett and R. Power. “State of the Art: Context Management”. Tech. rep., M-Zones Research Programme, in Ambient Intelligence: European Conference, AmI , 2008, Germany.
[6] C. Bettini, O. Brdiczka, K. Henricksen, J. Indulska, D. Nicklas, A. Ranganathan, and D. Riboni. “A Survey of Context Modelling and Reasoning Techniques”. Pervasive and mobile computing, 6(2), 2010, p.p. 161–180.
[7] A.K. Dey, “Understanding and using context”. Personal and Ubiquitous Computing, 5(1), 2001, p.p. 4–7.
[8] R. Shojanoori, R. Juric, and M. Lohi, “Computationally Significant Semantics in Pervasive Healthcare”, Journal of Integrated Design and Process Science, 16(1), 2012, p.p. 43–62.
[9] R. Shojanoori, “Towards Formalisation of Situation Specific Computations in Pervasive Computing Environments”. PhD thesis, 2013, University of Westminster, London, UK.
[10] Y. Yang, L. Wu, G. Yin, L. Li, and H. Zhao,”A Survey on Security and Privacy Issues in Internet-Of-Things. IEEE Internet of Things Journal, 4(5), 2017, p.p. 1250–1258.
[11] I. Butun, P. Österberg, and H. Song. Security of the Internet of Things: Vulnerabilities, Attacks, and Countermeasures. IEEE Communications Surveys & Tutorials, 22(1), 2019, p.p. 616–644.
[12] J.L. King, “Centralized Versus Decentralized Computing: Organizational Considerations and Management Options”. ACM Computing Surveys (CSUR), 15(4), 1983, p.p. 319–349.
[13] K. Ren, A. Thomson, and D.J. Abadi, “An Evaluation of the Advantages and Disadvantages of Deterministic Database Systems”. Proceedings of the VLDB Endowment, 7(10), 2014, p.p. 821–832.
[14] S. Rafaeli and D. Hutchison, “A Survey of Key Management for Secure Group Communication”. ACM Computing Surveys (CSUR), 35(3), 2003, p.p. 309–329.
[15] Y. Jung, E. Festijo, and M. Peradilla, “Joint Operation of Routing Control and Group Key Management for 5G ad-hoc D2D Networks”, IEEE International Conference on Privacy and Security in Mobile Systems (PRISMS), 2014, p.p. 1–8.
[16] B. Schneier, “Applied Cryptography: Protocols, Algorithms, and Source Code In C”. John Wiley and Sons, 2007.
[17] A.J. Menezes, P. C. Van Oorschot, and S.A. Vanstone, “Handbook of Applied Cryptography”, CRC Press; 1st edition (16 Oct. 1996).
[18] N.D. Jorstad and T.S Landgrave, “Cryptographic Algorithm Metrics”, Institute for Defense Analyses Science and Technology Division, 1997, available at https://csrc.nist.gov/csrc/media/publications/conference-paper/1997/10/10/proceedings-of-the-20th-nissc-1997/documents/128.pdf
[19] D.J. Bernstein, “Introduction to Post-Quantum Cryptography. In Post-Quantum Cryptography, Springer, 2009, p.p. 1-14.
[20] T. Kothmayr, C. Schmitt, W. Hu, M. Brünig, and G. Carle, “Dtls based Security and Two-Way Authentication for the Internet Of Things” Ad-Hoc Networks, 2013, 11(8), p.p. 2710–2723.
[21] B. Daghighi, L.M. Kiah, and S. Afkhami Rad “Group Key Management Using Public Key Exchange”, International Journal of Computer Theory and Engineering, 5(5), 2013) p.p. 807-8011.
[22]. A. Lyth, ”Towards context based decision making for Group Key Management (GKM) in an IoT Setting”, MSc Thesis, University of South Eastern Norway, June 2021.
[23] H. Ma and G. Sun, “Blockchain-based Group Key Management Scheme in IoT”, In International Conference on Intelligent Computing, Springer, 2020, p.p. 445–457.
[24] K. Surani and M. Dhamecha, “Internet of Things Based Block-chain Key Management”, In 2019 International Conference on Smart Systems and Inventive Technology (ICSSIT), 2019.
[25]V.S. Naresh, S. Reddi, and N.V.E.S. Murthy, “A Provably Secure Cluster Based Hybrid Hierarchical Group Key Agreement for Large Wireless Ad Hoc Networks” Human-centric Computing and Information Sciences, 9(1), 2019, p.p.1–32.
[26] C. Esposito, M. Ficco, A. Castiglione, F. Palmieri, and A. De Santis,“Distributed Group Key Management For Event Notification Confidentiality Among Sensors”, IEEE Transactions on Dependable and Secure Computing, 17(3), 2018, p.p.566–580.
[27] A.B. Feroz Khan and G Anandharaj, “A Cognitive Key Management Technique for Energy Efficiency and Scalability in Securing The Sensor Nodes in the IoT Environment: CKMT” SN Applied Sciences, 1(12), 1575, 2019.
[28] K.E. Nolan and L.E. Doyle, “Teamwork and Collaboration in Cognitive Wireless Networks”, IEEE Wireless Communications, 14(4), 2007, p.p. 22–27.
[29] O.C. Moholth, R. Juric, and K. Moholth McClenaghan, “Detecting Cyber Security Vulnerabilities through Reactive Programming” In the 52 Hawaii International Conference on System Sciences, HICSS 52, HI, US, 2019.
[30] R Juric and I Kim, “Software Architectures for Smart Applications which Merge Ontological Reasoning With Big Data Analytics” In the 22nd International SDPS 2017 conference, AL, US, 2017.
[31] R Juric, “Could Semantic Web Technologies Create New Computational Models outside Semantic Web” In the 21st SDPS conference, FL, US, 2016.