Universidad Carlos III de Madrid

e-Archivo
Institutional Repository

This is a postprint version of the following published document:

Cuevas, A. & Bouckerche, A. (2013). Improving security in data-centric storage for wireless sensor networks. *PE-WASUN '13 Proceedings of the 10th ACM symposium on Performance evaluation of wireless ad hoc, sensor, & ubiquitous networks*, pp: 129-136. Barcelona: Association for Computing Machinery (ACM).

"© 2013 ACM. This is the author's version of the work. It is posted here for your personal use. Not for redistribution. The definitive Version of Record was published in *WASUN '13 Proceedings of the 10th ACM symposium on Performance evaluation of wireless ad hoc, sensor, & ubiquitous networks*, https://dx.doi.org/10.1145/2507248.2507252"

Improving Security in Data-Centric Storage for Wireless Sensor Networks

Ángel Cuevas[*][†] Azzedine Bouckerche[†]

ABSTRACT

This paper proposes a novel mechanism to provide with se-curity to existing Data-Centric Storage (DCS) solutions for Wireless Sensor Networks. The goal is to achieve a high security level without modifying standard DCS premises or increasing the network overhead. This means we just use the messages an operations already defined by DCS solu-tions. Our goal is to fulfil two security requirements: (*i*) only legitimate nodes for an application should be able to access the information of that application, and (*ii*) avoid long-term Denial of Service attacks targeting an application that operates in the network. Toward this end we define two different solutions depending on whether the sensor nodes in the network are resource-limited or powerful. We run exten-sive simulations and discuss the efficiency of the proposed solution under two different DCS solutions: GHT that pro-poses to use a single and static storage node per application, and STARR-DCS that uses multiple storage nodes per appli-cation that in addition change over the time. Based on the obtained results and discussion we conclude that changing replication nodes over the time is by itself a smart approach to avoid long-term attacks.

WSN, Data-Centric Storage (DCS), Security.

1. INTRODUCTION

Wireless Sensor Networks (WSNs) have been studied for a decade now and are slowly being exploited in real applications. In the classical literature WSNs are defined in a hierarchical way where the sole mission of sensor nodes is to monitor external parameters and send them to a central base station that will be in charge of processing them and take decisions. However, with the advance on the study of WSNs we can also find self-managed WSNs where no central node is required to allow the network to operate, so sensor nodes are in charge of storing and processing the informa-tion and take decisions in a distributed manner avoiding the necessity of a powerful central base station. An example of these networks are the so-called unattended sensor networks [7, 23, 22] that are deployed in remote areas (e.g. dessert, volcano, jungle, etc) with difficult human-access and prob-lems to establish long distance communications to remotely monitor the WSN. Furthermore, these self-managed WSNs, where sensors need to communicate to each other and take decisions, can be seen as a first step in the area of Internet of Things (IoT). Similarly to the sensor nodes in self-managed WSNs, IoT devices create networks where they communi-cate to each other and have necessity to store and process the information and take decisions based on the processed information.

We can find a large catalogue of sensor nodes (and IoT devices) going from very low-resource nodes (and very cheap) to very powerful nodes (and much more expensive as compared to the previous ones) that include powerful Operative System that bring them close to being a small computer instead of the traditional concept of a sensor mote. These powerful nodes are much more versatile and allow to program complex algorithms that require high storage and processing capacities, but eliminates the traditional paradigm of dealing with very limited but very cheap devices, whose goal is to use many of these devices (tens, hundred or even thousands) to complete complex tasks. In this paper, we will take into account both type of sensors and propose a solution according to their features.

A third aspect that we need to discuss in the introduction is what is the application scope of a WSN. Generally, WSNs are understood as application oriented, that is, a WSN is designed to implement a single task (i.e. run a single application). Although it may be reasonable that sensor nodes implement a single task at the application layer, there are many basic and common tasks that can be shared by sen-sor nodes in the network belonging to different applications. For instance, sensor nodes could route packets for nodes that are participating in a different application. In addition, sen-sor nodes could act as storage nodes for data that belongs to a different application, so the only associated task is to reply back that information in case the node is queried for

[*]Department of Telematic Engineering
Universidad Carlos III de Madrid, Spain
acrumin@it.uc3m.es

[†]PARADISE Research Lab
SITE - University of Ottawa, Canada
boukerch@site.uottawa.ca

it. Under these conditions it would be feasible to deploy large-scale WSNs where different companies share their sensors for basic tasks getting the advantage of distributing the deployment cost to cover large areas.

Therefore, the context of this paper is on self-managed WSNs composed by a large number of sensor nodes belonging to different applications but sharing common tasks such as routing and storage.

In this context Data-Centric Storage (DCS) [19] appears as a promising solution. DCS was introduced as a novel distributed information storage and delivery mechanism in which a node is selected as a rendezvous point to store all the events of a particular application. Then, for instance, one node in the network stores all temperature events, another node humidity events, another one fire events, etc. All nodes producing information of a particular application (i.e. producer nodes) compute the rendezvous node (named home node) for that application, and store its information on it. In turn, a node that wants to retrieve information from that application (i.e. consumer node) just needs to query the home node that replies with the application's stored data. The beauty of DCS is that each node locally computes which is the home node for a particular application, and by using the underlying routing layer can store/retrieve data on/from that home node.

The fact of using a single rendezvous point could lead to overload that node in case of a high demanding application (e.g. many temperature measurements stored in the node and many queries to retrieve the temperature information stored in the node). To alleviate this problem we proposed STARR-DCS in [2, 3] in which several home nodes (or replicas) are selected at random for only a limited amount of time. After that time a new set of home nodes is selected for the next period. In order to change the home nodes over the time we divide the time in periods called *epochs*. Then, during an epoch all application events are stored in the selected home nodes. As demonstrated in [2] these solution prolongs the network lifetime and distribute the network load across all nodes in the network.

Using DCS (either with a single home node or with multiple replicas) under the proposed context presents serious security concerns that need to be addressed. We are targeting networks where several applications (potentially from different companies) are running on top of the same WSN taking the advantage of sharing network resources, including storage. This implies that a sensor node of APP_i could be storing its events in nodes of APP_j, where sensors of APP_i and APP_j belongs to different companies. Furthermore, if none security mechanism is in place, any illegitimate node could easily retrieve information for whatever application in the network by just computing which are the home nodes for that application. Even more, in case the goal of a malicious node is to degrade de performance of a particular application, it just needs to detect one (or more) rendezvous nodes for an application and overload them by sending a large amount of queries, thus avoiding legitimate nodes for that application to access the stored information. We are meaning that unless some security solution is provided performing Denial of Service (DoS) attacks in DCS is straightforward.

Therefore, in this paper we want to propose a solution to completely avoid or at least reduce as much as possible the next three security problems:

1. A malicious node inserted in the network can access information from an application it does not belong to.

2. A sensor node of APP_j storing information of APP_ican access that information.

3. A malicious node inserted in the network performs a DoS attack to degrade the performance of an application running on the WSN.

In this paper we propose two different solutions depending on the sensor nodes capacity. Our goal for both solutions is to avoid the necessity of extending DCS paradigm with new messages or functionalities. First we propose a very simple solution that alleviates security issues 1 and 3 mentioned above in case STARR-DCS is in place. In contrast, DCS solutions that keep home nodes position static over the time appears as an easy target for malicious nodes that after some time can compromise an application. In addition, when sensor nodes are powerful, we propose a second solution based on Public-Key Cryptography. This solution solves the two first security issues above, but by itself cannot solve the DCS attack unless it is applied in a dynamic DCS network where home nodes of an application change over the time.

The remainder of this paper is structured as follows: Section 2 presents a background on DCS. Section 3 is the core piece of this paper where we introduce our proposal to provide with security DCS solutions. We present related works in Section 4 and conclude the paper in Section 5.

2. DCS BACKGROUND

In this section we present the functionality of the original Data Centric Storage proposal [19] as well as our extension that uses multiple replicas that change over the time, referred as STARR-DCS (Spatio-Temporal Adaptation of Random Replication for Data Centric Storage) [2][3].

2.1 Basic DCS

Ratnasamy *et al.* [19], first defined the concept of Data Centric Storage (DCS). They combined the idea of a Distributed Hash Table (DHT) [20, 13, 16] together with the Greedy Perimeter Stateless Routing (GPSR)[12], a geographic routing protocol, to create a DCS system called Geographic Hash Table (GHT).

In order to employ geographic routing, this work assumes that sensors are able to locate themselves within the sensornet by using GPS or any other location device or system. In addition, the size and borders of the network are well-known by all nodes.

In GHT when a producer sensor detects an event, it uses a hash function over the application or event name (e.g., *hash('APP')*). The hash function provides as the output some spatial coordinates inside the sensor field. Then, when a producer detects an event, it gets the spatial location provided by the hash function and invokes a *put(k,d)* operation, where k (e.g. *'APP')* following the key for the hash function above) is the key for the event type and d the data that forwards the data towards that spatial location using GPSR. The closest node to that spatial location becomes the home node for that event type and receives the producer message, because GPSR itself is enough to find the closest node to a given position. In turn, when a consumer wants to retrieve the data related to that event type, it uses the same hash function over the event type, and thus it obtains exactly the same spatial location. Next, it uses a *get(k)* operation that forwards a query using GPSR to that spatial location, thus reaching the home node that replies with the stored data for that event type.

2.2 STARR-DCS

The main problem of GHT is that it proposes to use a single rendezvous or home node, which under high load applications will become saturated in a short time in terms of storage and processing, and what is more important it will run out of battery quickly.

Several papers in the literature [15] (from GHT authors) [9][4][17] propose to use a uniform placement of multiple rendezvous nodes (referred as replicas or replications nodes throughout the paper) to alleviate the load of a single home node. The number of replicas selected directly impacts on the network load, which in turns affects the network lifetime. The main problem of these approaches is that they fail on selecting an appropriate number of replicas. For instance, [15] and [9] need to use as number of replicas(N_r) equal to 4, 16, 64, 256, etc (i.e. follows a formula $N_r = 4^d$, where d=1, 2, 3, 4, etc). In many cases none of the possible values is suitable, but a value in between should be used instead.

STARR-DCS[2, 3] proposes to locate replicas at random in the network. This provides more flexibility in the number of replicas selected, since the application can select whatever number. The results presented in [2] shows that Random Replication improves all previous approaches in terms of minimizing the network load.

Furthermore, all previous solutions rely on static replicas. That is, they compute N_r positions for the replicas that do not change. Therefore, the closest node to each position is selected as rendezvous node until it runs out of battery. Then the next closest node is selected until it runs out of battery, and so on. This implicitly leads to the creation of big routing and sensing holes in the network that are very harmful for the network operation as reported in [2]. To solve this problem, STARR-DCS proposes to change replicas over the time. That is, a set of nodes will serve as replicas for a given time window known as *epoch*. When the epoch expires a new set of random nodes will be selected, and so on. This dynamic solution efficiently distributes the energy expenditure across the network avoiding the big holes problem that appears in static solutions and effectively extends the network lifetime in at least 60% as reported in [2]. Finally, STARR-DCS offers a framework that includes all the protocols and mechanisms required to be implemented in real nodes.

Finally, we want to introduce the function used by STARR-DCS to compute the location of the replicas which is necessary to understand the solution proposed in this paper. STARR-DCS uses a very simple function, which is an adaptation of the hash function used by GHT to compute the home node. In STARR-DCS all nodes know its application key (k), the current epoch (e) and the number of replicas being used N_r in that application. With this information any node is able to compute the rendezvous points' locations at any particular time. For that, a node just needs to compute the following hash operation: $hash(k \oplus e \oplus i), \quad \forall i \in [1, N_r]$ that generates N_r random locations within the network. The closest nodes to those locations serve as rendezvous nodes.

3. SOLUTIONS FOR SECURING DATA CENTRIC STORAGE

In this section we present our proposal to solve the security problems that exist when DCS (both in GHT and STARR-DCS) is applied in self-managed WSNs with sensors of different applications sharing network resources. As we described in the introduction we aim at providing simple yet efficient solutions according to the sensor nodes capability. In addition, we want to propose a solution that just uses the premises already defined for DCS, that is the use of put and get operations without any further extension. With this goal in mind we want to achieve or at least approach the next security concerns:

1. A malicious node inserted in the network can access information from an application it does not belong to.
2. A sensor node of APP_j storing information of APP_i should not be able to access that information.
3. A malicious node inserted in the network performs a Denial of Service (DoS) attack to degrade the performance of an application running on the WSN.

Let us start defining what is the current security of GHT and STARR-DCS in case an illegitimate node want to exploit DCS vulnerabilities to compromise the security of an application.

In order to find the home node in GHT the malicious node just needs to know the key (k) of the targeted application either to retrieve information or to perform a DoS attack to degrade the application's performance. Therefore, all the security in GHT depends on how easy is to derive the key of an application. Several DCS solutions propose straightforward mechanism to define k such as using the name of the application itself, or assigning sequential key values to the different applications running in the network. Hence, in these cases a malicious node could very easily jeopardize an application.

In the case of STARR-DCS there are 3 input parameters used in the hash function to produce the coordinates of the replicas: the key (k) that identifies the application, a value i that goes between 1 and the number of replicas (N_r), and the epoch identifier (e). STARR-DCS is a bit more secure than GHT due to the use of e. First of all, i does not provide any security since is a mere sequence going from 1 to N_r, and it may be enough to query one of the replicas since depending on the operation mode they may store the same information [3]. Nonetheless, inferring e is not straightforward since it could be an arbitrary number imposed by the application (not necessarily following a sequential order). However, STARR-DCS defines a non-secure mechanism (named Meta-Information Service) to easily retrieve the value e for any application running in the network. Thus, a malicious node could use that mechanism to get the value of e. Then once a malicious node knows k and e can find all replica nodes and query them to get the information. Although we could propose a solution to provide security to the mechanism to get the value of e, this would be a solution just oriented to STARR-DCS, and we want to find a more general purpose solution applying to any DCS system.

In this paper we assume that a malicious node could get k and e and use the common hash function to all network nodes and compute the location of any replication node location. In addition, we also assume that an attacker cannot infer the destination coordinates by overhearing the channel and analyzing the routing header. Security at the routing layer is out of the scope of this paper.

Finally, we must notice that depending on the sensor nodes capacity we could provide more or less secure solutions. Therefore, our goal is to provide a two different solutions: a first one for resource-constrained sensor nodes, and a second one for powerful sensor nodes.

3.1 Resource-constrained sensor nodes Solution: Hiding rendezvous nodes position

In a network where sensor nodes are very resource-limited we cannot implement complex security mechanism (e.g. cryptography) since it would not be supported by sensor nodes. Then, if we assume that nodes can just use those functionalities required for DCS solution, we can propose a solution that increments the security level by just making more difficult to find where the rendezvous nodes are located.

When a malicious node wants to attack a particular application it just simply needs to identify which are the nodes acting as rendezvous nodes for that application (or the unique home node in case of GHT) and query them. Assuming, that the malicious node knows the application key, k (that would be enough in GHT), and the epoch id, e, the malicious node could easily perform an information retrieval or DoS attack.

In order to make the attack more difficult, we propose to include a new parameter s in the hash function which is only known by the application nodes. This parameter will be different for each application and all nodes of an application will be pre-programmed with the value of s for that application. Hence, only nodes of that application will be able to find the rendezvous node for that application at any time. Therefore, in order to find the nodes serving as replicas for a particular application an attacker would need to implement a force brute attack generating random locations and querying them, so that the closer node to that location would respond in case it is in charge of the application with key k that is being targeted by the malicious node.

The success time of the attack will be established (among other parameters) by 2 parameters that are especially interesting for us: the number of nodes in the network and the number of replicas being used for the application. Next we perform extensive simulation experiments and discuss the results for each of these parameters

3.1.1 Number of nodes in the network

Figure 1 presents simulations results showing what is the average number of queries (over 5000 experiments) a malicious node would need to perform to success (i.e. find the actual home node) in a network with a single home node for different number of sensor nodes randomly deployed in a network covering a square area of 1000x1000 m^2. The number of nodes in the network varies from 100 to 1000 using a step of 100.

As the number of nodes in the network is increased the attacker needs to send more queries to reach the valid home node. In particular we see a linear growth with the number of nodes in the network. It is important to notice that even for the smallest number of nodes evaluated (i.e. 100 nodes), the attacker would need to issue 153 queries, which means it will query half of the nodes twice before finding the correct node. If we think on some kind of distributed and collaborative misbehaviour approach it would be easy identifying and filtering the attacker. In addition, if we analyze more populated networks the results show that for a network with 500 nodes the attacker would need to launch 730 queries before succeeding, and in case of having 1000 nodes in the network this number would be close to 1400 queries in average.

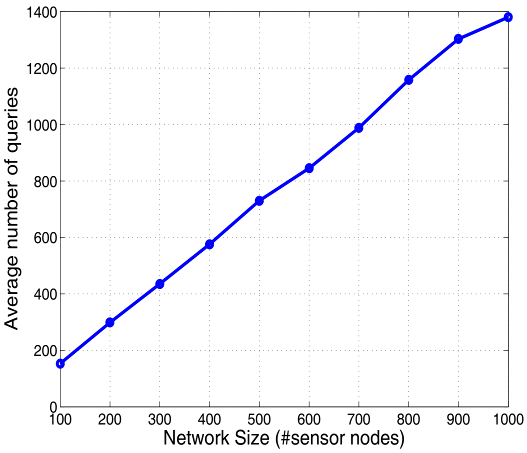

Figure 1: Number of queries issued by a malicious node to succeed in an attack to retrieve information from a particular application with respect to the number of nodes deployed in the network.

It must be noted that in the networks we are targeting (self-managed WSNs) attackers will be battery-powered nodes as well. Therefore, the more energy they expend before compromising the home node, the less remaining energy they have to successfully complete their attack. Therefore, in terms of security countermeasures, when we are not able to fully avoid the attack it is important to provide solutions that reduces the harmful effects as much as possible. In this case, the difference between having 100 or 1000 nodes in the network implies that the attacker needs to expend roughly 10x more energy to succeed in the attack.

3.1.2 Number of rendezvous nodes

We analyze separately the case when the attacker aims to retrieve information and the case when its goal is to perform a DoS attack.

Illegitimate information retrieval.

As we introduced in Section 2 we can find several solutions in the literature (including STARR-DCS) that propose to use several replication nodes to store the content. The worst case from the security point of view is when all the replication nodes are storing all the information for a particular application. In such case as soon as the attacker compromise one replica it will get access to all the information related to the targeted application. In this subsection we analyze the influence that the number of replication nodes have in terms of security for this worst case.

Figure 2 presents the average number of queries issued for an attacker when 1, 2, 3, 4 or 5 replicas are used to store the data produced by a particular application. For this evaluation we have employed a 1000x1000 m^2 network

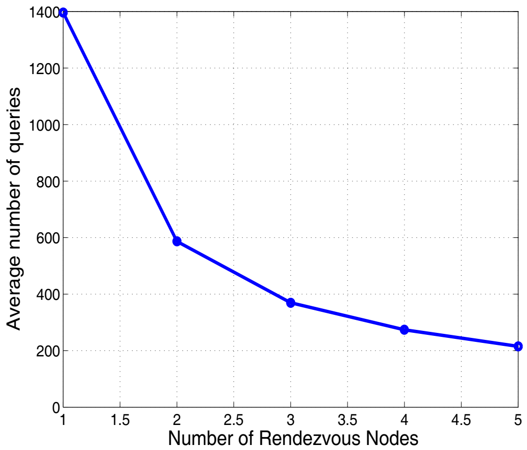

Figure 2: Number of queries issued by a malicious node to succeed in an attack to retrieve information from a particular application with respect to the number of replication nodes used to store applications' information.

where 1000 nodes have been randomly deployed. We have averaged the number of queries out of 5000 experiments.

The graph clearly shows that using several replication nodes is quite harmful in terms of security when we cannot use any more sophisticated security mechanism (e.g. cryptography). For instance, when we move from a single home node to 2 replication nodes the attacker needs to send around 2.5x less queries to succeed in its attack, and this number is reduced more than 6x in case the application uses 5 rendezvous nodes.

Denial of Service attack.

In this case to perform a DoS attack the fact of having multiple replicas helps on delay the attack success. In order to fully avoid the operation of an application the attacker needs to identify all the replicas and overload them so that neither producers node (that store new application data in the rendezvous nodes) nor consumer nodes (that query replicas to retrieve the stored information) can normally operate. For instance, in STARR-DCS when a node fails accessing one replica it tries sequentially to access the rest of replicas in that epoch until one of them is available. Therefore, if the attacker does not disconnect all the replicas it will downgrade the application's performance, but all the sensors of that application will be still able to operate.

Figure 3 shows the average number of queries that an attacker needs to issue to fully disconnect an application for different number of replicas. We have used the same network parameters than in the previous experiment.

As we expected the attacker will need more time to compromise replication nodes as they increase in number. Due to the use of s in the hash function a malicious node will need to randomly generate queries until it finds all N_r replication of nodes. The number of queries issued grows linearly with the number of nodes.

3.1.3 Discussion for GHT and STARR-DCS

We want to discuss what is the real security improvement achieved by this solution for resource-limited nodes in the two systems we are using as reference in this paper: GHT that uses a single and static home node, and STARR-DCS that uses multiple rendezvous nodes that change over the time.

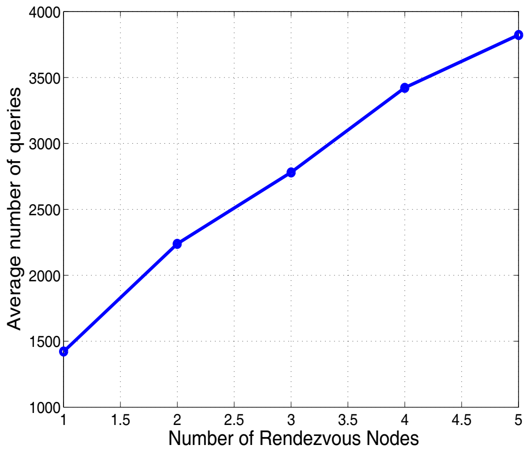

Figure 3: Number of queries issued by a malicious node to succeed in a DoS attack to fully avoid an application operating in the network..

GHT.

Based on the results the fact of using a single home node is worthy from the security point view for those attacks that wants to retrieve application information. However, from the moment the attacker discovers the position of the home node the security is broken and the attacker will be able to access the application information at any time. That is, once it finds some coordinates that bring its queries to the current home node (even though they may not be exactly the same coordinates produced using the parameter s in the hash function), the attacker already knows a quite close position to the exact coordinates. Therefore, when the home node runs out of battery and the next closest node to the hash output coordinates becomes the home node, the attacker will need very few queries to find the new home node. It does not need to generate random locations again. This weakness is due to using a static position for home nodes. Therefore, following this reasoning DCS multi-replication approaches that propose to use static coordinates for the rendezvous nodes would be weaker for the information retrieval attack due to the use of multiple replication nodes (as shown in Figure 2).

In the case of the DoS attack for GHT the reasoning is the same. As soon as the malicious node finds the home node location and have enough querying capacity to overload the home node, it can perform a DoS attack until its battery expires. In the case of multiple static replicas a full-DoS attack is more costly since the attacker will need more time to identify all the replication nodes, but once these nodes are identified the attacker can perform a continuous full-DCS attack that avoids the operation of the targeted application.

In a nutshell, if the attacker takes 20 minutes (e.g. 1200 queries sent once per second) to find the home node, from that moment on, it will be able to access the information for the targeted application and/or perform a DoS attack by overloading the rendezvous node.

STARR-DCS.

Based on the results the main weakness of STARR-DCS is the fact that it proposes to use multiple replicas, and this is initially a bad decision for the information retrieval attack (i.e. the attacker wants to gather information from the targeted application). However, it has a strong characteristic from the security point of view based on the fact that STARR-DCS changes the replicas over the time, and every time an epoch expires, a new set of replication nodes are selected at random. This means that even if the attacker succeeds in a particular epoch, once that epoch expires the attacker will need to start another brute force attack to find the position of some of the new rendezvous nodes. Therefore, in this case assuming that we can estimate the maximum query rate from an attacker we could easily protect our application by just tuning the epoch duration. For instance, let us assume we want to protect an application that is using 3 rendezvous nodes in a network with 1000 nodes that covers an area of 1000x1000 m^2. In addition, we predict that the maximum query rate of an attacker based on the type of nodes working in the network is 1 query per second. From Figure 2 we know that an attacker needs 350 queries in average to find at least one of the three rendezvous nodes and access the desired information. This means that if we establish an epoch duration of 300 seconds (i.e. 5 minutes) the attacker would not be able to find the current rendezvous nodes in time. But even more important, if the attacker is lucky and in just few queries finds one of the targeted application rendezvous nodes, it will have access to the information only for a limited amount of time.

Furthermore, the fact of changing the rendezvous nodes over the time eliminates the possibility of a long-term DoS attack since the attacker will be able to intensively load the attacked rendezvous nodes just for a limited amount of time. Therefore, in contrast to the static DCS solutions, STARR-DCS appears as a good solution to avoid DoS attacks because it changes the replication nodes over the time.

In summary, although the proposed mechanism does not guarantee that the attacker will not be able to retrieve the targeted application information, it establishes a good security solution for resource-limited sensor nodes that are not able to use cryptography mechanisms. This solution is very simple and does not add any overhead or complexity to DCS and make much harder the potential success of the attacker. In addition, we have discussed that changing the position of the rendezvous over the time does not only provides better load balancing, and a longer network lifetime [2], but also considerably improves the security and avoids long-term DoS ttacks.

We must notice that the proposed solution for low-resource sensor nodes covers requirements 1 and 3 (see list above), but does not help to solve the second point. The only solution for that is to use some cryptography mechanism that avoids the storing node to access the information it is storing from other applications.

3.2 Powerful Sensor Nodes Solution: Public-key Cryptography

The previous solution does not provide full guarantee that the attacker will not access sensible information. Therefore, while it is a reasonable solution under the premises established for resource-constrained nodes, it would be a bad solution when sensor nodes are powerful. In this case we can provide a solution that fully ensures the access to the application data to only authorised nodes.

The solution is as simple as using Public-Key Cryptography. Each application will be assigned a Public-Private key pair. These keys are pre-programmed in the sensor nodes running the application. The Public-Key Cryptography works as follows, the sender uses the Public-Key to encrypt the message it want to send to the receiver. It must be noted that the public key, as its name indicates, can be used by any user. In turn, the receiver of the information uses the Private-key (that only she knows) to decrypt the message.

The Public-Key Cryptography would be mapped to the DCS paradigm as follows. A particular application has producer nodes that generates the information and send it to the rendezvous nodes, and consumer nodes that query the rendezvous nodes to retrieve the stored information. Then, we propose that producer nodes use the public key to encrypt the application information before sending it to the rendezvous node. In this way, the rendezvous node will store only the encrypted data. In turn, any sensor node in the network will be able to access the rendezvous node and retrieved the stored data. However, only the consumer nodes that posses the private key will be able to decrypt the information gathered from de rendezvous node.

Therefore, a malicious node can retrieve the data for a particular application but it will be unable to decrypt it.

This solution works for any DCS solution independently on whether it uses a single or multiple nodes, or whether the rendezvous nodes are static or change over the time. In addition, it does not introduce any overhead since we still rely on the simple put and get operations defined by GHT. The only overhead is in processing to encrypt/decrypt data.

This solution is valid for the information retrieval attack since neither illegitimate nodes nor nodes storing data for other application will be able to access the information. Therefore, this solution fulfils requirements 1 and 2, but not 3 as it is defined.

In order to prevent long-term DoS attack this solution needs to be combined with a DCS solution that changes the replicas over the time, such as the case of STARR-DCS.

4. RELATED WORK

The scope of this paper is security for Data-Centric Storage in Wireless Sensor Networks.

We have already introduced and referenced previously in the paper the main DCS solutions relevant for this paper. For a broader view of DCS field we recommend [4].

From the security side, researchers in the area have made a great effort that have led to a large variety of works for WSNs security in the literature. We can find surveys like [21] that presents a quite complete overview of security in WSNs, or [11] which presents a specific security analysis for secure routing in WSNs. In addition, we can find specific security solution at all different layers. For instance, authors in [10] propose a full link layer security architecture. We can also find secure routing protocol proposals for WSNs like SNEP or μTesla [14]. Finally, security issues in WSNs are also studied at the application layer. For example, authors in [1] analyze security issues in WSNs for healthcare applications. Furthermore, in our paper we propose the use Public-key Cryptography. Aligned to this we can find quite a few works in the literature that studies the use of Key Cryptography in WSNs [6][5][8].

In spite of the comprehensive work for both areas in WSNs, DCS and security, we could just find one previous work merging them in order to provide with security to DCS solutions, which is named pDCS [18]. As we do in this paper this solution tries to propose a solution to avoid illegitimate users to access non-allowed information, but it does not address DoS attacks. They propose to use a set of keys that depends on location and time. Although the solution is robust it presents several drawbacks as compared to our proposal. First, it relies on a DCS solution that divides the network into grids, thus they do not talk about storage/rendezvous nodes anymore, but about storage cells. This makes their solution only valid for those DCS that performs such type of division, which is not standard. In contrast, our solution is valid for all type of DCS networks independently on whether they perform network division into cells or not. Second, in their solution (as in the one we propose) the attacker generates random locations to try to compromise the storage cell for a particular application. The fact of relying on cells facilitates a lot the brute force attack as compared to the case when the attacker needs to generate actual coordinates location. This happens because the number of cells in the network is a lot lower than the potential coordinates. Therefore, relying on actual coordinates instead of cells for storage increases the difficulty of the attack. Third, pDCS requires the use of temporal keys, that is the key used to store information in a particular storage cell changes over time, thus the authors assume sensors in the network are synchronized and all of them are able to update the key at the same time. However this is a too strong assumption since synchronization in WSNs is a very complex issue that requires lot of overhead and energy consumption to be solved. Finally, pDCS is a static solution that does not take into account the necessity of changing the storage cell for a particular application over the time, which, as we have discussed in the paper, is essential to avoid DoS attacks.

5. CONCLUSIONS

In this paper we have discussed security aspects in the field of Data Centric Storage for Wireless Sensor Networks. Our goal was to provide a security mechanism that: (*i*) avoid (or at least complicate) the access of illegitimate nodes to information that is restricted to nodes belonging to a particular application, and (*ii*) avoid the possibility of a long-term Denial of Service (DoS) attack that totally or partially impacts the operation of a particular application in the network. Toward this end we have presented two different solutions that takes into account whether the sensors are resource-constrained or powerful. On the one hand, we have presented a first solution for low-resource sensor which is quite powerful for DCS solutions that have multiple rendezvous nodes per application that change over the time, like STARR-DCS. This solution considerably limits the access of malicious nodes to illegitimate information and in addition avoids DoS attacks. On the other hand, we have introduced a solution based on Public-Key Cryptography suitable for powerful sensor nodes with enough capabilities to perform encryption/decryption functions. With this solution only those nodes belonging to an application can access that application's information, however this solution by itself does not solve DoS attack. Finally, we have learnt that changing home nodes over the time not only reduces network load and extends network lifetime (as we demonstrated in previous works), but it has very positive impact on security, since implementing this dynamism by default avoids long-term attacks.

6. AKCNOWLEDGEMENT

The research leading to these results has been partially funded by the Spanish MEC under the CRAMNET project (TEC2012-38362-C03-01) and by the General Directorate of Universities and Research of the Regional Government of Madrid under the MEDIANET Project (S2009/TIC-1468), NSERC DIVA Network Program, Canada Research Chairs Program, the Ontario Research Fund (ORF), ORNEC, the Early Ontario Researcher Award, the Ontario Centres of Excellence (OCE).

7. REFERENCES

[1] M. Ameen, J. Liu, and K. Kwak. Security and privacy issues in wireless sensor networks for healthcare applications. *Journal of Medical Systems*, 36(1):93–101, 2012.

[2] A. Cuevas, M. Urueña, and G. de Veciana. Dynamic random replication for data centric storage. In *Proceedings of the 13th ACM international conference on Modeling, analysis, and simulation of wireless and mobile systems*, MSWIM '10, pages 393–402, New York, NY, USA, 2010. ACM.

[3] A. Cuevas, M. Urueña, G. de Veciana, and A. Yadav. STARR-DCS: Spatio-Temporal Adaptation of Random Replication for Data Centric Storage. *ACM Transactions on Sensor Networks*, To be published Ferb, 2014.

[4] A. Cuevas, M. Urueña, R. Romeral, and D. Larrabeiti. Data centric storage technologies: Analysis and enhancement. *Sensors*, 10(4):3023–3056, 2010.

[5] W. Du, J. Deng, Y. Han, S. Chen, and P. Varshney. A key management scheme for wireless sensor networks using deployment knowledge. In *INFOCOM 2004. Twenty-third AnnualJoint Conference of the IEEE Computer and Communications Societies*, volume 1, pages –597, 2004.

[6] W. Du, J. Deng, Y. S. Han, and P. K. Varshney. A pairwise key pre-distribution scheme for wireless sensor networks. In *Proceedings of the 10th ACM conference on Computer and communications security*, CCS '03, pages 42–51, New York, NY, USA, 2003. ACM.

[7] D. Estrin, R. Govindan, J. Heidemann, and S. Kumar. Next century challenges: scalable coordination in sensor networks. In *Proceedings of the 5th annual ACM/IEEE international conference on Mobile computing and networking, MobiCom '99*, pages 263–270, New York, NY, USA, 1999. ACM.

[8] X. Fan and G. Gong. Lpkm: A lightweight polynomial-based key management protocol for distributed wireless sensor networks. In J. Zheng, N. Mitton, J. Li, and P. Lorenz, editors, *Ad Hoc Networks*, volume 111 of *Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering*, pages 180–195. Springer Berlin Heidelberg, 2013.

[9] Y.-J. Joung and S.-H. Huang. Tug-of-war: An adaptive and cost-optimal data storage and query mechanism in wireless sensor networks. In *Proceedings of the 4th IEEE international conference on Distributed Computing in Sensor Systems, DCOSS '08*, pages 237–251, Berlin, Heidelberg, 2008. Springer-Verlag.

[10] C. Karlof, N. Sastry, and D. Wagner. Tinysec: a link layer security architecture for wireless sensor networks. In *Proceedings of the 2nd international conference on Embedded networked sensor systems*, SenSys '04, pages 162–175, New York, NY, USA, 2004. ACM.

[11] C. Karlof and D. Wagner. Secure routing in wireless sensor networks: attacks and countermeasures. *Ad Hoc Networks*, 1(2?Äì3):293 – 315, 2003. <ce:title>Sensor Network Protocols and Applications</ce:title>.

[12] B. Karp and H. T. Kung. GPSR: greedy perimeter stateless routing for wireless networks. In *"Proceedings of the sixth annual international conference on Mobile computing and networking, Mobicom '00"*, pages 243–254, New York, NY, USA, 2000. ACM.

[13] P. Maymounkov and D. Mazières. Kademlia: A peer-to-peer information system based on the xor metric. In *First International Workshop on Peer-to-Peer Systems, IPTPS '01*, pages 53–65, London, UK, 2002. Springer-Verlag.

[14] A. Perrig, R. Szewczyk, J. D. Tygar, V. Wen, and D. E. Culler. Spins: security protocols for sensor networks. *Wirel. Netw.*, 8(5):521–534, Sept. 2002.

[15] S. Ratnasamy, B. Karp, S. Shenker, D. Estrin, R. Govindan, L. Yin, and F. Yu. Data-centric storage in sensornets with GHT, a geographic hash table. *Mobile Network Applications*, 8(4):427–442, 2003.

[16] A. Rowstron and P. Druschel. Pastry: Scalable, decentralized object location, and routing for large-scale peer-to-peer systems. In *Lecture Notes in Computer Science*, pages 329–350, 2001.

[17] R. Sarkar, X. Zhu, and J. Gao. Double rulings for information brokerage in sensor networks. In *Proceedings of the 12th annual international conference on Mobile computing and networking, Mobicom '06*, pages 286–297, New York, NY, USA, 2006. ACM.

[18] M. Shao, S. Zhu, W. Zhang, G. Cao, and Y. Yang. pdcs: Security and privacy support for data-centric sensor networks. *Mobile Computing, IEEE Transactions on*, 8(8):1023–1038, 2009.

[19] S. Shenker, S. Ratnasamy, B. Karp, R. Govindan, and D. Estrin. Data-centric storage in sensornets. *SIGCOMM Computer Communication Review*, 33(1):137–142, 2003.

[20] I. Stoica, R. Morris, D. Karger, M. F. Kaashoek, and H. Balakrishnan. Chord: A scalable peer-to-peer lookup service for internet applications. In *Proceedings of the ACM SIGCOMM Conference, SIGCOMM '01*, San Diego, California, August 2001.

[21] Y. Wang, G. Attebury, and B. Ramamurthy. A survey of security issues in wireless sensor networks. *Communications Surveys Tutorials, IEEE*, 8(2):2–23, 2006.

[22] C.-M. Yu, C.-Y. Chen, C.-S. Lu, S.-Y. Kuo, and H.-C. Chao. Acquiring authentic data in unattended wireless sensor networks. *Sensors*, 10(4):2770–2792, 2010.

[23] J. Zhang, M. Walpola, D. Roelant, H. Zhu, and K. Yen. Self-organization of unattended wireless acoustic sensor networks for ground target tracking. *Pervasive Mobile Computing*, 5(2):148–164, 2009.