A FRAMEWORK FOR TRUSTED WIRELESS SENSOR NETWORKS

Joon S. Park and Abhishek Jain

Abstract Wireless sensor technologies have become increasingly important in a variety of areas, including mission-critical applications. However, there are still many obstacles that we need to overcome before we apply the current wireless sensor network (WSN) technologies. Unlike traditional computing devices in a wired environment, the WSN faces significant challenges from wireless node devices, which usually have limitations in computational power, energy, and their working environment. In this paper we introduce a framework for a trusted large-scale WSN that provides a longer sensor lifetime, cost effectiveness, security, survivability, and scalable management. Our framework uses clustering mechanisms and multiple cluster heads within a cluster where we switch the cluster heads based on their current energy levels or security/survivability reasons. We also present the Extended Random Key Pre-distribution scheme adapted for such a framework.

1. INTRODUCTION

Remote sensing technologies have become increasingly important in a variety of areas, including medical, business, military, and geological applications. Simultaneously, we are witnessing a rapid growth in the availability and deployment of wireless devices. The opportunity exists to apply the principles of sensor networks to power-constrained wireless devices, greatly enhancing their capabilities through ad-hoc deployment and aggregation of available resources. Wireless sensor networks (WSN) are being seen as the medium that will connect the Internet with the physical world in the near future. However, many obstacles need to be overcome before we apply the current WSN technologies to mission-critical applications. When sensor networks and wireless communication with ad-hoc characteristics and dynamics are combined, trustworthiness challenges increase exponentially. Sensors may collaborate to process and send data to a processing center. Unlike traditional computing devices in a wired environment, the WSN faces significant challenges from the nature of the devices and their working environments. Those constraints can be summarized as follows.

- Limitations in availability of electrical power
- Low processing power and memory
- Ad hoc network (joining and leaving numerous network environments as the devices move)
- Low bandwidth
- Possible physical damage

Despite the above-mentioned limitations, a trusted environment is provided for the sensors by means of adaptive and self-configurable management, especially for a mission-critical application, such as target tracking in a tactical area or patient monitoring in a hospital. In this paper, we provide a cluster-based framework for trusted large-scale WSNs. We believe our approach will provide more robust WSN services with longer sensor lifetime and greater cost effectiveness, security, survivability, and scalable management.

For simplicity, in this paper, we describe our approach using two kinds of sensors: a set of basic sensors with limited computational power and energy constraints, but are cheap, and a set of more powerful sensors that are capable of being a cluster head (we call them cluster-head-capable nodes), but are more expensive. However, our approaches can also be used for a more heterogeneous WSN. Researchers have been concentrating for the past few years on various issues that are critical to the energy-efficient communication in sensor networks. Most of these works focused on homogeneous sensor networks with a flat level-topology. Much less attention has been given to hierarchical topology of sensor nodes. In a hierarchical topology, sensor nodes are grouped to form a cluster, and some nodes are given special privileges and responsibilities to represent the clusters. Such a node is called cluster head.

Generally, a cluster-based WSN consists of multiple clusters and can provide more time and energy-efficient, scalable services compared to direct transmission among the nodes (node-to-node communications). Moreover, clustering has the potential to make the network more secure and detection of malicious nodes easier. In a dynamic WSN that supports nodes's mobility, if the node-IDs are not unique, there is a possibility that multiple sensors with the same ID can join the same cluster, which may cause an operational problem. Therefore, in a dynamic WSN, even regular sensors should have unique IDs. In cluster-based topology, time synchronization is required only within clusters, while in direct transmission topology, it is required in the entire WSN. Furthermore, a cluster head performs data aggregation or first-level analysis after it collects data from the regular nodes in the cluster. This allows the application to exploit correlation for compression and data fusion and may increase the accuracy and performance of the WSN. In our approach, we also propose multiple cluster-head-capable nodes in each cluster. There are two reasons why we propose multiple cluster heads in a cluster. First, a cluster head has more serious energy constraints because it consumes more power than regular sensors. Second, if there is only one cluster head in each cluster, one cluster head's anomaly (because of failure or cyber attacks) affects the entire cluster. If this happens in a mission-critical application, the anomaly may affect the high level mission of the WSN. Therefore, we propose multiple cluster-head-capable nodes in each cluster, providing more energy-efficiency, survivability, security, and scalable management. As we will discuss later, we propose to switch the cluster head in each cluster for more energy efficient, secure, and survivable communications. However, this also introduces new challenges such as dynamic configuration and cryptographic key management. We will also discuss how we can solve these new problems based on our approaches.

2. RELATED WORK

2.0.1 Clustering the Sensor Nodes. Ghiasi et al. [8] discuss the theoretical aspects of the clustering problem in sensor networks. These aspects are specific with application to energy optimization. The authors present an optimal algorithm for clustering the sensor nodes such that each cluster (which has a master, namely, a cluster head) is balanced and the total distance between sensor nodes and master nodes is minimized. Bandyopadhyay et al. [3] describe a distributed randomized clustering algorithm for WSNs. The algorithm is used to form a multilevel hierarchy of cluster heads in the network. The authors' simulation results show that the energy consumption decreases as the hierarchy among the cluster heads increases. The fact behind the result is suggested to be the increased "in-network processing" of the data generated by the sensors before submitting it to the base station. Krishnamachari1 et al. [10] present a self-organizing algorithm that combines shortest-path routing mechanisms with leader election to permit nodes within each region to self-organize into routing clusters. This scheme results into a multi step, self-organizing and energy efficient solution for extraction of information about environmental features. Krishnan et al. [11] proposed two algorithms that produce clusters of bounded size and low diameter. The goal of the algorithms is to reduce the message complexity for energy and bandwidth considerations.

2.0.2 Cluster Heads. The distributed clustering algorithm (DCA [1]) is suitable for networks with static nodes or those with a very low mobility. The algorithm elects nodes to be cluster heads based on the weights associated with the nodes. These weights are generic and can be defined based on the application. The cluster head is selected as the node with the highest weight among its 1-hop neighbors. The distributed and mobility-adaptive clustering algorithm (DMAC) is a modification of the DCA algorithm that will allow node mobility during or after the cluster set-up phase [2].

Chandrakasan et al. [9] proposed LEACH as an energy-efficient clustering protocol for wireless sensor networks accomplished by switching cluster heads. The authors were able to demonstrate an improvement of the lifetime of the sensor network by 8 times. One critical point that differentiates our scheme from LEACH is that, in LEACH, the nodes themselves decide, if they want to be a cluster head, while in our scheme the nodes are initially selected to be a cluster head by the base station, and later on by the previous cluster head or the base station (if the previous station cannot function correctly). Moreover, LEACH considers a purely homogeneous sensor network that has limitations, especially for a large-scale WSN, compared with the cluster-based approach we described in introduction. Additionally, LEACH proposes direct communication from a cluster head to the base station, which might be more energy consuming as well as infeasible sometimes, while we propose multiple hops between a cluster head and the base station. As each cluster head is a normal sensor node in a homogeneous topology, it has some communicational range restraints, which creates a high probability that a cluster head would not be able to communicate with the base station if it is out of its range. The situation is very critical and general to large-scale wireless sensor networks in which the base station is too far away from the sensor nodes.

2.0.3 Multicasting vs. Unicasting. Deb et.al. [6] propose one-hop broadcasting to be more reliable and energy efficient than unicasting in wireless sensor networks. Broadcasting transmits information packets without the address of a particular destination, in contrast to unicasting, in which information packets contain the address of one and only one intended destination. Broadcasting is generally used when some information is intended to be processed by everyone or when the source doesn't know the address of the intended destination. Typically, broadcasting is more reliable than unicasting but, at the same time, has some inherent drawbacks associated with it. Even in one-hop broadcasting, packets are multiplied that are processed by an extra number of nodes, which consume their energy for no valid reason. Additionally, it increases traffic in the network. We propose minimum broadcasting in sensor networks to minimize extra computation by the sensor nodes. We propose a direct communication (unicast) between a sensor node and its cluster head, and a cluster head to another cluster head. Note that information assurance can still be achieved by using acknowledgments from the destination. For example, data forwarded from a node to its cluster head that is not very critical can be transmitted without using acknowledgments, while critical data delivery is followed by corresponding acknowledgments.

3. SYSTEM ARCHITECTURE

In this section we highlight the system architecture of our approaches. Since we consider a large-scale WSN in this paper, we will use a heterogeneous and clustered WSN, based on the reasons we described in the previous sections.

3.0.4 Node Communications. We assume that basic sensors have been deployed in the target area in a non-deterministic way. However, the special nodes which are going to act as cluster heads are deployed deterministically so that all the nodes have at least one cluster head. There are base stations in an operational environment. These base stations coordinate to perform various functions for the network. Also these base stations are powerful machines with highly computational communication capabilities, and no energy constraints.

There is no direct communication between the basic nodes. All communications go through the cluster head in each cluster. In contrast to other proposed schemes where node-to-node communication is allowed, our scheme facilitates only node to cluster head communication. Therefore, the main communication mechanism in our scheme is unicasting. Multicasting or broadcasting may be used rarely by the base station or a cluster head in situations such as reconfiguration or publishing a common message to its nodes. This approach aids in bringing more energy efficiency to the network, as unicasting will not consume the energy of a number of extra nodes other than the destination. Forwarding a packet through broadcasting would result in some processing and possibly multiplication of packets by other nodes, as well. Each and every node would communicate through its cluster head (the cluster head of the cluster to which that node belongs). The cluster head is then responsible for forwarding the message to the destination (in fact, to the next level in the path). The cluster head chooses its next hop at the time when it is selected and all the keys are established. In any event, the cluster head forwards the message either to the base station or to any of the other cluster heads, whichever is the next hop. If the packet is forwarded to any other cluster head, it repeats the same procedure to make the message reach to the base station ultimately. In a nutshell, we have communications between:

- Node and cluster head
- Cluster heads
- Cluster head and base station

The basic sensor nodes in the network may have different functions, based on the applications. The cluster heads are sensor nodes with strong communication and computational capabilities, but still have energy constraints. These powerful nodes are assumed to be costlier than basic nodes. The cluster heads communicate with each other, with basic nodes, and with base stations. Basically, all the communications from node to cluster head and between cluster heads use unicasting. However, multicasting and acknowledging techniques can be used for data delivery assurance or announcement (e.g., a cluster head's declaration in its cluster).

3.0.5 Cluster-Based Heterogeneous Sensor Networks. For simplicity, a set of homogeneous devices (i.e. all the nodes have the same computing power) may be considered for a small-scale WSN, where the network topology can be either flat-level or multi-level. Flat-level topology supports node-to-node communications, while hierarchical topology supports cluster-based communications. Many literatures have discussed issues related to homogeneous WSNs, especially the flat-level topology. However, some applications need a large-scale WSN, which consists of a large number of wireless sensors with different functions. Although the homogeneous approach could work technically for a large-scale WSN, it is not a reasonable approach in terms of cost-effectiveness, scalability, energy-efficiency, security, and survivability. Therefore, we introduce a framework for trusted large-scale, cluster-based, and heterogeneous WSNs. Although there can be multiple levels of nodes in our WSN topology, we describe just two different kinds of nodes (cluster-head-capable nodes and basic nodes). Basic nodes whose main functions include sensing, monitoring events, processing data and sending it to special nodes act as cluster heads. These nodes may have different computational and communicational capabilities, based on their missions. Cluster heads are special nodes with extra computational and communicational capabilities. These nodes are deployed (more than one in our approach) in each cluster of basic sensor nodes.

4. OPERATIONAL SCENARIO

4.0.6 Pre-Deployment. In the pre-deployment stage, we store some specific information (e.g. a set of keys, node identities, cluster identities, application specific functions, etc.) in all the nodes, which are then deployed in the target area. Clusters are formed among the nodes during this pre-deployment stage using various techniques mentioned in Section 2.0.1. Depending on the application, nodes might be assigned specific identification according to the tasks they are going to perform once they deployed. Basically, all the information that would be needed to form and manage clusters is loaded in sensor nodes before deployment. Additionally, some secret information is also assigned to each sensor node that would be needed while establishing keys in the network. For example, a set of keys with a corresponding set of key identifiers could be stored in each node. This set of keys can then be used later to establish pair-wise keys with the cluster head.

4.0.7 Deployment. In a WSN, node deployment can be done in either a deterministic or non-deterministic manner. Purely deterministic deployment makes things very easy, but is not so reasonable for a large-scale WSN. Obviously, it is not scalable. Large number of sensors makes deterministic deployment impossible for most systems. On the contrary, purely non-deterministic deployment provides too loose control for cluster formation, configuration, and coverage area. Therefore, in this paper we introduce a hybrid approach, in which, basic sensor nodes are deployed randomly (non-deterministically) in the target area, but certain special nodes are deterministically deployed to act as cluster heads for clusters of sensor nodes.

Once the nodes are pre-loaded with information, they are deployed in the target area. The sensor nodes are then partitioned into clusters and the cluster-head-capable nodes are deployed in each cluster. Initial cluster heads were chosen for each cluster in the pre-deployment step.

- Formation of clusters:

  The formation of a cluster can be application-specific. The application might require sensor nodes to be grouped together according to the tasks they are assigned at the time of pre-deployment. The base station could partition the nodes into clusters according to the density of the sensor, or it could use the location information of sensor nodes to divide them into clusters. Various kinds of techniques are proposed to partition a sensor network into clusters (summarized in Section 2.0.1).

- Selection of initial cluster heads:

  It is obvious that cluster heads will do much more processing and will consume their energy at a greater rate than the other normal nodes in the cluster. For that reason, in our approach cluster heads are chosen and switched periodically on the basis of the energy or security levels of the nodes with cluster head capability. Note that initially, as all the cluster-head-capable nodes have the same energy level, the base station simply picks up the most optimal node as the cluster head. For example, the optimal node could be the center node in the cluster.

One concern while deploying wireless nodes in a WSN would be the even distribution of those nodes within the target area, so that each and every node is in reach of at least one cluster head. Furthermore, the cluster heads are not crowded with a large number of sensor nodes to be managed. If we are deploying a sensor network in a controllable and reachable environment such as an emergency room of a hospital, this issue is not a big concern, because we can manually distribute cluster heads all along the network fairly. However, the problem arises when the sensor network is deployed in an uncontrollable area, such as a battlefield or forest, where the nodes are simply thrown into the area by an airplane or some other medium. Although our scheme would work perfectly for the first case, we assume that at least the cluster-head-capable nodes could be deployed deterministically in the second case.

4.0.8 Operational Modes. In our approach there are two modes of switching a cluster head: ***restart*** and ***continue*** [13]. Each of these methods has its pros and cons. A restart mode interrupts all the sessions going on in the network, while a continue mode preserves the state information of all the sessions going on. In other words, in restart mode all the sessions are started from scratch, while a continue mode carries on with the sessions going on when the cluster heads are being switched. Considering the applications of the sensor network, it seems that switching the cluster heads in restart mode is not feasible. For instance, say a sensor network is monitoring the signs of a probable earthquake. The decision is taken after a great deal of data gathering and processing by the sensor nodes. If somehow, to switch the cluster heads in the network, this session is interrupted and all the information is lost, we could fail to guess the event of the upcoming earthquake. This defeats the whole purpose of deploying the sensor network to monitor seismological events.

The above example illustrates the need to switch the cluster heads in continue mode rather than in restart mode. However, switching a cluster head in a continue mode also brings new issues to be addressed such as dynamic key establishment and session transfer. Nevertheless, restart mode is required when the network is deployed or reconfigured. For instance, if the secret key of a current cluster head has been compromised, we need to switch the cluster head and restart the session with a new secret key.

4.0.9 Cluster Head Switching in Continue Mode. The cluster-head-capable nodes in a cluster are switched to be an actual cluster head from time to time, depending on the current energy level of the nodes or other security/survivability reasons. A node, on being selected as a cluster head, will broadcast its authority of cluster head of the cluster to a specified area (cluster). All the nodes belonging to that area will then establish a pair-wise key with their cluster head (described in Section 5), which will then be involved in the process of forwarding data from the basic nodes of its cluster to the base station. As it goes below some energy level threshold or has other problems, it begins the process of selecting another cluster head and finally gives up its authority to a new cluster head. If the current cluster head cannot function correctly to make this transition, the base station will be in charge of the cluster head switching.

Two different modes, namely, restart and continue mode, are described for the component survivability in distributed systems in our previous work [13]. Here we consider switching cluster heads in continue mode. For some applications it might be necessary to switch cluster heads without affecting the session going on in the network. For instance, say a cluster head is involved in the process of fusion of data being received from two nodes and going below some threshold of energy level, which might make it necessary to switch itself with some other cluster-head-capable node with sufficient energy to be cluster head. In such a case, the present cluster head is involved in the session transition. There are two possible solutions: either interrupt the session and start it from the beginning with the new cluster head (restart mode), or preserve the processing of the current session with the intermediate results, which are handed over to the new cluster head, continuing the session with the new cluster head (continue mode). The restart mode is straightforward. We consider how a cluster head can be switched in a continue mode as follows.

In continue mode, the cluster head publishes its resignation in its cluster and requests energy levels of other cluster-head-capable nodes in its cluster. After receiving the energy level of all cluster-head-capable nodes, the current cluster head picks up the node with the highest energy level and notifies it tht it will be the new cluster head. The current cluster head also sends all of the session information it is involved in at the present time. This information includes the session ID, interest, nodes involved in it, and the intermediate results. The present cluster head also notifies each of the involved nodes in the session to send their results to the new cluster head.

For example, say a cluster head has been involved in three data fusion processes of the data received from nodes A and B for an interest I5, from nodes M, N, and P for interest I3 and from nodes A, P, and D for an interest I2. Suppose all the basic nodes are monitoring the temperature of a certain region. The session information sent to the new cluster head, in this case, would be something like the following table:

Table 1. **Session Information**

Session Information	*Concerned Interest*	*Nodes Involved*	*Intermediate Results*
Data Fusion process (aggregation)	I5	A,B	44.67F
Data Fusion process (aggregation)	I3	M,N,P	67.87F
Data Fusion process (aggregation)	I2	A,P,D	54.89F

In the continue mode, the cluster head also notifies each of the nodes in its cluster about the new cluster head. Each node, on receiving this information establishes a new pair-wise key with the new cluster head in the same manner as it did with the previous cluster head (described in Section 5). The nodes involved in the same session with the previous cluster head will use the same key with the new cluster head (unless the key is not compromised), which they were using with the previous one, until the session is completed. If the new cluster head does not have some of those keys (being used by the nodes involved in the session) in its key pool, it can get them from the old cluster head through a secure communication channel so that the nodes do not need key establishment again. A detailed key establishment scheme is described in Section 5. As soon as the transition is finished, the previous cluster head loses its authority as cluster head. The new cluster head then can continue this session with the nodes involved.

5. EXTENDED RANDOM KEY PRE-DICTRIBUTION

Key distribution is the starting procedure for most security services. Authentication and secure communications are based on the keys distributed in a secure manner. There are many possible schemes for this purpose, but, a secure and scalable key distribution is a challenging problem in WSNs because of their limitations. A traditional security framework for powerful wired devices, such as public key cryptography, is not suitable for WSNs. Technically, current WSNs do not have computational power and memory space for public key cryptography. Alternatively, we can use secret key cryptography for the same purpose. However, it is not always possible to apply pre-distribution of secret keys among the nodes in the same cluster in a large-scale WSN, which needs non-deterministic deployment.

Our key establishment mechanism is adapted from the basic scheme of Random Key Pre-distribution with our extensions. A fair amount of work has been done to use the scheme in the context of sensor networks [5][4]. However, most of the existing approaches have been applied to node-to-node communications, while we need a cluster-based topology for a large-scale WSN. Node-to-node communications require a large amount of key space in sensor nodes. The requirement becomes further complicated when the number of sensor nodes increases in the network and ultimately leads to a scalability problem. In this work, therefore, we attempt to apply the Random Key Pre-distribution Scheme to cluster-based sensor networks, in which we don't have node-to-node but only node-to-cluster head communication, to solve the scalability problem. Additionally, keys are established for direct paths between nodes and cluster heads and between cluster heads. In case a cluster head does not have any matching key with a node in its cluster, a matching key is found among other nodes with which a secure link is already established. This matching key is then imported to the cluster head key space so that it can communicate with the node directly rather than through an indirect multi-hop path, as is done in the original Random Key Pre-distribution Scheme. As multi-hop paths are avoided, this modification makes the scheme even more energy efficient.

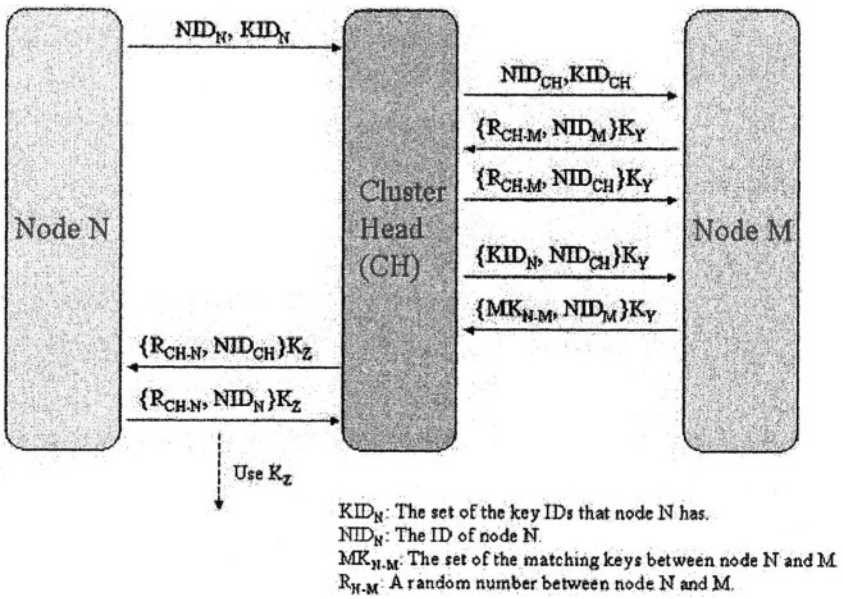
Node N
Cluster Head (CH)
Node M
NID_N, KID_N
NID_{CH}, KID_{CH}
$\{R_{CH\text{-}M}, NID_M\}K_Y$
$\{R_{CH\text{-}M}, NID_{CH}\}K_Y$
$\{KID_N, NID_{CH}\}K_Y$
$\{MK_{N\text{-}M}, NID_M\}K_Y$
$\{R_{CH\text{-}N}, NID_{CH}\}K_Z$
$\{R_{CH\text{-}N}, NID_N\}K_Z$
Use K_Z
KID_N: The set of the key IDs that node N has.
NID_N: The ID of node N.
$MK_{N\text{-}M}$: The set of the matching keys between node N and M
$R_{N\text{-}M}$: A random number between node N and M.

Figure 1. Key Establishment with the Help of Other Nodes

Each sensor node in the network is pre-deployed with a set of keys randomly picked from a large key pool, say P.

$P = k_1, k_2, k_3..........k_n$

$k_1, k_2, ...k_n$ are the keys in the pool *P* where n is a large number. This key pool is then used to pick up several subsets, $P_1, P_2, ...P_m$. such that

$P = P_1 \bigcup P_2 \bigcup P_3 \bigcup .... \bigcup P_m$, where m is the total number of nodes deployed in the network.

Each sensor node (including the regular nodes and cluster-head-capable nodes) is assigned one of these random subsets of *P*. A sensor node uses its set of keys to establish a pair-wise key with its cluster head. Since each cluster head keeps only a subset of the entire key pool *(P),* for security and scalability reasons, it might be the case that there is no common key in the key sets of the node and the cluster head. In this case, the cluster head finds a common key with the help of other nodes with which it already has established a secure channel. How the cluster head establishes a key in such a situation is depicted in Figure 1. In particular, it shows how node N and cluster head *CH* can establish a shared secret key securely with the help of node *M*.

Figure 1 shows a cluster in a sensor network with one current cluster (*CH*) head and two other regular nodes. Based on probability, we assume that there are some regular nodes (at least one, such as *M* in the figure) that have a common key with *CH*. When *N*, which does not have any matching key with *CH*, requests a key establishment to *CH* by sending its *ID* (denoted by NID_N in the figure) and the set of the key *ID*s that *N* has (denoted by KID_N in the figure), the *CH* and *N* cannot discover a matching key initially. *CH* then generates a secure connection with another node (*M* in the figure) by following the key establishment procedures described above. After it generates a secure channel with *M*, *CH* forwards KID_N (that it received from *N*) to *M*. The node *M* finds all of the matching keys (denoted by MK_{N-M} in the figure) in its key set (e.g., P_2) with *N*'s key *ID*s and sends those matching keys to *CH* via a secure channel. This is possible because *CH* and *M* have matching keys in their key pools. Suppose *CH* and *M* have chosen a shared secret key K_Y, as shown in the figure. The set of matching keys MK_{N-M} (between *N* and *M*) and *M*'s identity NID_M are encrypted with K_Y and the encrypted information is sent to *CH*. *CH* then decrypts the encrypted information using K_Y and picks up one of the keys out of MK_{N-M}, say K_Z, and establishes it as a pair-wise key with N as described in the previous case.

In case *M* does not have any matching key with *N*, other nodes are tried in the cluster. Unfortunately, if none of the regular nodes in the cluster can provide a matching key with *N* to *CH*, the *CH* then requests other cluster-head-capable nodes that have at least one matching key with the requesting *CH*, in the cluster or even in different clusters. In our example, before the *CH* forwards *N*'s key identities (not real keys) to *M*, *CH* and *M* should have at least one matching key. If no node has the key *CH* is looking for, or if there is no secure channel between *CH* and the node who has the matching key, the *CH* can get the key ultimately from the base station, which keeps a complete key set of the sensor network. The cross-cluster key transfer is useful when a new node is joining from a different cluster.

6. SURVIVABILITY

Sensors are small wireless devices with limited power capability. In a real world scenario, thousands of sensors can be deployed and connected to the grid network. Information is gathered from all the sensors and sent to the grid for complex computations and analysis. It is very important to have the sensors up and running all the time for mission-critical services (e.g. military applications). Therefore, we need to provide survivability to some mission-critical wireless sensor networks.

Survivability is the capability of an entity to continue its mission even in the presence of damage to the entity [14]. An entity ranges from a single component (e.g. a sensor in our case), with its mission in a distributed computing environment to an information system that consists of many components to support the overall mission. An entity may support multiple missions. Damage can be caused by internal or external factors such as attacks, failures, or accidents. To make a system survivable, it is the mission of the system, rather than the components of the system, to survive.

We categorize the models into three types, namely, static, dynamic, and hybrid models. The static model is based on redundant entities, prepared before the operation, to support critical services continuously to the client in an operational environment. Typically, implementing the static model is simpler than implementing the dynamic model. However, it does not provide the immunization capability that the dynamic model does. In the dynamic model, the entities, which caused failures are with failures, or are under attack, are replaced by dynamically generated components on the fly and deployed (in runtime) by the factory as and when they are required. In our case, for instance, if one sensor node does not provide its mission correctly because of cyber attacks or internal failure, a survivable WSN should provide another sensor node that can take the affected sensor's place. Detailed comparisons between the static and dynamic models and descriptions about the hybrid model are provided in [13].

Sensors are subject to service (mission) failure because of physical damage or software failures. Analyzing the reason for the failures and providing immunization is a challenging research area in this case. Physical recovery is limited because of the restricted location of the sensors and the possibility of physical damage to the sensor. If the failed sensor is physically damaged, then the practical solution is to have redundant empty nodes (sensors) and deploy the software needed for the sensor on the fly, based on our dynamic survivability model. If the damage is not physical (due to software failure), then overwriting the software on the sensor is a possible solution.

7. CONCLUSION

In this paper we introduced a framework for trusted WSNs, providing a longer sensor lifetime, cost effectiveness security, survivability, and scalable management. Our framework uses clustering mechanisms and multiple cluster heads within a cluster where we switch the cluster heads based on their current energy level or for security/survivability reasons. This approach makes a large-scale WSN more robust, but it also brings new requirements such as cryptographic key management and transparent session transition. Therefore, we also described how to satisfy those requirements in terms of different operational modes and dynamic key establishment between the sensor nodes.

Acknowledgments

This work was supported in part by the New York State Center for Advanced Technology in Computer Applications and Software Engineering (CASE).

References

[1] S. Basagni. *Distributed Clustering for Ad Hoc Networks,* in Proceedings of International Symposium on Parallel Architectures, Algorithms and Networks, pp. 310-315, June 1999.

[2] S. Basagni. *Distributed and Mobility-Adaptive Clustering for Multimedia Support in Multi-Hop Wireless Networks,* in Proceedings of Vehicular Technology Conference, Vol. 2, pp. 889-893, 1999.

[3] Seema Bandyopadhyay and Edward J. Coyle. *An Energy Efficient Hierarchical Clustering Algorithm for Wireless Sensor Networks,* IEEE INFOCOM 2003.

[4] Haowen Chan, Adrian Perrig and Dawn Song. *Random Key Predistribution Schemes for Sensor Networks,* IEEE Symposium on Security and Privacy, 2003.

[5] Laurent Eschenauer and Virgil D. Gligor. *A Key-Management Scheme for Distributed Sensor Networks,* the 9th ACM Conference on Computer and Communication Security, Washington D.C., November 2002.

[6] Budhaditya Deb, Sudeept Bhatnagar and Badri Nath. *Information Assurance in Sensor Networks,* the 2nd ACM international conference on Wireless sensor networks and applications, San Diego, CA, September 2003.

[7] Stefan Dulman, Lodewijk v. Hoesel, Tim Nieberg, Paul Havinga. *Collaborative communication protocols for wireless sensor networks.*

[8] Soheil Ghiasi, Ankur Srivastava, Xiaojian Yang, and Majid Sarrafzadeh. *Optimal Energy Aware Clustering in Sensor Networks,* invited paper, sensors 2002, 2, 258-269, July 12th 2002.

[9] Wendi Heinzelman, Anantha Chandrakasan, and Hari Balakrishnan. *Energy-Efficient Communication Protocols for Wireless Microsensor Networks,* Proc. Hawaaian Int'l Conf. on Systems Science, January 2000.

[10] Bhaskar Krishnamachari 1 and S. Sitharama Iyengar. *Efficient and Fault-tolerant Feature Extraction in Wireless Sensor Networks,* 2nd Workshop on Information Processing in Sensor Networks, IPSN, Palo Alto, California, April 2003.

[11] Rajesh Krishnan, David Starobinski. *Message-Efficient Self-Organization of Wireless Sensor Networks,* IEEE WCNC 2003, New Orleans, March 2003.

[12] Seapahn Meguerdichian1, Sasa Slijepcevic 1, Vahag Karayan, Miodrag Potkonjak1. *Localized Algorithms in Wireless Ad-Hoc Networks: Location Discovery And Sensor Exposure,* ACM Symposium on Mobile Ad Hoc Networking & Computing (MOBIHOC), Long Beach, CA, Oct. 4-5, 2001, pp. 106-116.

[13] Joon S. Park and Pratheep Chandramohan. *Component Recovery Approaches for Survivable Distributed Systems,* the 37th Hawaii International Conference on Systems Sciences (HICSS-37), Big Island, Hawaii, January 5-8 2004.

[14] Joon S. Park and Judith N. Froscher. *A Strategy for Information Survivability,* 4th Information Survivability Workshop , Vancouver, Canada, March, 2002.

[15] Hairong Qi, Xiaoling Wang, S. Sitharama Iyengar. *Multisensor Data Fusion in Distributed Sensor Networks Using Mobile Agents,* Proc. Intl. Conf. Information Fusion, pp. 11-16, August 2001.

[16] Tilak, Abu-Ghazaleh and Heinzelman. *A Taxonomy of Sensor Network Communication Models,* ACM Mobile Computing and Communication Review.