Securing Wireless Sensor Networks: Introducing ASLAN - A Secure Lightweight Architecture for WSNs

Michael Collins[1], Simon Dobson[2], Paddy Nixon[2]
Systems Research Group, School of Computer Science and Informatics,
UCD Dublin, Ireland

[1]*michael.collins@comp.dit.ie*
[2]*{simon.dobson, paddy.nixon}@ucd.ie*
[1]*http://www.comp.dit.ie/mcollins*
[2]*http://www.csi.ucd.ie/Staff/AcademicStaff/{sdobson, pnixon}*

***Abstract*—Wireless sensor networks consist of many small, inexpensive devices that have constraints in coverage, bandwidth, storage resources, communications ability and processing power. Therefore security issues are a critical concern due to possible exposure to malicious activity and potential threats. As a result of the physical constraints in sensor nodes, traditional cryptographic techniques are not suitable to operate on such networks where security requirements are of crucial importance. This raises serious concerns on finding methods to protect sensor nodes from adversaries, to quickly segregate those that have been attacked, and allow the network to reform. To address the security vulnerabilities in a wireless sensor network, this paper proposes a secure lightweight architecture (ASLAN) that takes account of the constraints of sensor networks. With the aid of a base station, a hierarchical network topology is formed allowing end-to-end communication between sensor nodes. ASLAN also supports identifying and isolating aberrant sensor nodes.**

Keywords—Wireless Sensor Network; Architecture; Security; Protocol

1. Introduction

Wireless sensor networks have emerged as a technology that are being quickly adopted due to their flexibility and use in a variety of environments. However, they consist of small, inexpensive devices or nodes that have severe constraints such as limited bandwidth, limited processing power, short battery life, small storage capability and are physically prone to external threats [1]. Even with all the advantages that wireless sensor networks provide such as fast deployment and configuration, the constraints of the sensor nodes makes them extremely vulnerable to various security threats [2]. These include attacks that target a specific node with endless communication in order to exhaust its limited battery life and also the physical vulnerability of the sensor nodes within a hostile environment, e.g., a military battlefield. Unfortunately, cryptographic techniques such as Public Key Infrastructure (PKI) [3], which is widely used in traditional wired networks, is not suitable to operate on sensor networks to enable secure data communication. Therefore, this makes sensor networks susceptible to attack and also very difficult to identify and deal with nodes that act maliciously.

In this paper, the authors propose a secure lightweight architecture for wireless sensor networks (ASLAN) that provides the desired security mechanisms to address the identified security threats. ASLAN employs the notion of a base station that is used as a base class in a hierarchical network configuration. The paper describes how this network topology is formed.

The structure of this paper is as follows: Section 2 discusses some of the various security issues associated with wireless sensor networks, Section 3 gives a summary of previously related research in this area, Section 4 introduces our proposed architecture (ASLAN) which includes the network topology formation, details of our security protocol for identifying and isolating aberrant nodes, and the secure routing mechanism of data communication in the sensor network, Section 5 discusses our future work to implement and evaluate ASLAN, and finally, Section 6 concludes this paper.

2. Security issues in Wireless Sensor Networks

Wireless sensor networks have innate constraints compared to traditional wired networks that prevent many security mechanisms being able to operate. This section gives examples of such constraints and discusses the threats and issues typically encountered.

2.1 Introduction

Traditional security mechanisms normally require high processing capability, and large memory and storage requirements. Such resources are not available in nodes in a wireless sensor network. As a result of these constraints, designing effective security mechanisms is more difficult than for a wired network. Examples of these constraints include:

(a) Small memory
The memory in a wireless sensor node is very limited memory with small storage capacity. As a result, any security mechanism to be designed and run within a sensor network will have limitations and not be as robust as one for a wired network.

(b) Reduced energy levels
Designing security mechanisms for wireless sensor networks must consider the reduced energy levels that are implicit with sensor nodes. When a sensor node is deployed, its energy source is usually a battery so it is critical to design security features that are not memory or power intensive in order to prevent the battery life being exhausted quickly. However, security features will consume extra energy that that required for normal operation, for example cryptographic techniques, and this may be detrimental to the sensor node's time to live.

(c) Communication problems
There is an inherent problem with wireless communication in that data can get intercepted, lost and is generally prone to attack. Since sensor nodes are usually deployed in mass numbers and form a sensor network, lots of data will be transmitted and received between sensor nodes resulting in heavy network traffic. This makes it likely that some data packets will be damaged or lost. Unlike traditional wired networks where protocols deal with such situations, the nodes in a wireless sensor network do not have the resources available to resend data packets.

(d) Physical security
Sensor nodes are generally small devices that are not very robust. This makes them prone to damage and vulnerable to attack in harsh and hostile environments where an attacker can potentially capture or damage a node. After a sensor node has been deployed, they are susceptible to issues such as weather conditions, undesired natural phenomena, deliberate attack by an adversary, and power exhaustion. It is almost impossible to have any control of these issues.

2.2 Threats and Issues in Wireless Sensor Networks

Most of the threats and attacks against security in wireless networks are almost similar to their wired counterparts while some are exacerbated with the inclusion of wireless connectivity. In fact, wireless networks are usually more vulnerable to various security threats as the unguided transmission medium is more susceptible to security attacks than those of the guided transmission medium [4].

Attacks can also be launched at any point in the network and that certain attacks may be more effective at different layers of the communications protocol. Table 1 depicts the various attacks that can be launched at different layers of the communications stack [5].

Table 1. Attacks on Communications Stack

Layer	Attack
Physical Layer	DOS – Jamming, Tampering Sybil
Data-link Layer	DOS – Collision, Exhaustion, Unfairness Interrogation Sybil – Data aggregation, Voting
Network Layer	DOS – Neglect & Greed, Homing, Misdirection (Spoofing), Black Holes, Flooding Sybil Wormhole Attack
Transport Layer	DOS – Flooding, De-synchronization

2.2.1 Denial of Service (DoS) attack

A DoS attack tries to exhaust the resources available to the victim node by sending unnecessary data packets and therefore prevents legitimate network users from accessing services or resources they desire [6, 7]. They take many forms as shown in Table 1 that prevent the network performing its expected functions. There are several types of DoS attacks in a wireless sensor network that include jamming, power exhaustion, service greed, and network flooding. Mechanisms that attempt to prevent a DoS attack may include payment for network resources, and robust authentication.

2.2.2 Sybil Attack

A Sybil attack is one in which a sensor node mimics the identity of more than one other legitimate nodes [8, 9]. The Sybil attack specifically targets situations where large tasks are divided into subtasks and distributed among several sensors in order to complete the task. The Sybil attack operates by attacking the distributed storage, routing mechanism, data aggregation, voting, fair resource allocation and misbehaviour detection of nodes [9]. All peer-to-peer networks are susceptible to a sybil attack. However, the detection of sybil nodes is difficult [9].

2.2.3 Sinkhole Attack

In a sinkhole attack, the adversary's goal is to lure nearly all the traffic from a particular area through a compromised node, creating a metaphorical sinkhole with the adversary at the center. Since nodes on, or near the path that packets follow have many opportunities to tamper with application data, sinkhole attacks can enable many other attacks (for example, selective forwarding) [10].

Sinkhole attacks operate by trying to attract network traffic and pass their data through a compromised node. For example, a compromised node could falsely advertise that it offers an efficient route from one point of the network to another. Due to either the real or imagined high quality route through the compromised node, it is likely each neighbouring node of the adversary will forward packets through the adversary, and also propagate the attractiveness of the route to its neighbours. Effectively, the adversary creates a large "sphere of influence", attracting all traffic from nodes several (or more) hops away from the compromised node [10].

One reason for mounting a sinkhole attack is that it offers an easy process for performing selective forwarding. Thus, most of the traffic in the vicinity of the compromised node will flow through it and the compromised node can then select whichever packets it desires to modify or suppress.

2.2.4 Wormhole Attack

A wormhole attack is one whereby an attacker tunnels messages received in one part of the network over a low latency link and replays them in a different part. A simple example of this attack is a single node situated between two other nodes forwarding messages between the two of them. However, they more commonly involve two distant malicious nodes colluding to understate their distance from each other by relaying packets along an out-of-bound channel available only to the attacker [11]. Wormholes may also be used simply to convince two distant nodes that they are neighbours by relaying packets between the two of them.

2.2.5 Attack on transit information

When sending data in a wireless sensor network, the information may be spoofed, modified, replayed or removed. An attacker can monitor the traffic being routed through the network and may interrupt, intercept, modify or fabricate data packets thereby sending inaccurate information to the recipient [12]. Due to the resource constraints of nodes, adequate security mechanisms to deal with such issues are difficult to implement.

There has been much research carried out to identify all the possible threats and issues to wireless sensor networks. However, these threats evolve over time with new security concerns emerging frequently that need to be addressed.

3. Related work

Research into security in wireless sensor networks has been conducted over recent years. This section summarizes some of this research.

Chen et al. [13] were among the early proposers of a security model for communication between a base station and the sensor nodes in a wireless sensor network. The model consists of two security protocols for the deployment of sensor networks. The first protocol is called "base station to mote confidentiality and authentication" and describes how an efficient shared-key algorithm be used to guarantee authenticity and privacy of information passing on the network.

The reason they use a shared-key algorithm is because of its low consumption of resources which is ideal for use on small, resource-constrained sensor nodes. The second protocol is called "source authentication" that implements a hash chain function to achieve mote authentication.

Perrig et al. [14, 15] proposed a model called SPINS which is a collection of protocols for sensor networks. It integrates SNEP (Secure Network Encryption Protocol) and μTESLA (micro-Time Efficient Streamed Loss-tolerant Authentication). SNEP supports end-to-end security by providing data confidentiality and two-way data authentication with minimum overhead. μTESLA, a micro version of TESLA, provides authenticated streaming broadcast and keeps computation costs low by using only symmetric cryptography.

However, the SPINS model leaves some unresolved security questions such as the security of compromised nodes, Denial-of-Service (DoS) issues, and network traffic analysis issues. Furthermore, this protocol assumes the static network topology ignoring the ad hoc and mobile nature of sensor nodes [16].

Undercoffer et al. [17] proposed a light weight security protocol operating in the base station of a sensor network framework. In this model, the base station can detect and remove a sensor node if it behaves anomalously or becomes compromised. However, there are no security measures specified on dealing with an attack such as the interception of communication between nodes.

Eschenauer et al. [18], proposed a key pre-distribution model where each sensor in the network receives a random subset of keys from a large key pool before they are deployed. This key pool is held by a base station. In order for communication to take place between nodes, a common key must be selected from each node's subset of keys and to use this as their shared key.

Chan et al. [19] extended the model in [18] in which they developed three key pre-distribution schemes; q-composite, multipath reinforcement, and random-pairwise keys schemes. Each of these schemes enabled the base station to pre-distribute keys to the nodes on deployment.

Du et al. [20, 21] introduced two different schemes. The first scheme proposed using pairwise key pre-distribution. Under this scheme, there would be a much higher payoff for an attacker to spend the large amount of time and resources required to compromise nodes in a large-scale sensor network than a smaller scale network. The second scheme proposed a key management mechanism whereby keys are issued to sensor nodes based on deployment knowledge which stores the position of sensors prior to their deployment. However, since neighbouring nodes must use the same key (symmetric cryptography) for communications, the problem exists in that there is no way to know the exact locations of neighbour nodes due to the randomness of node deployment. However, it is feasible to know a set of likely neighbouring nodes so the use of a random key pre-distribution technique is possible using [18].

Undercoffer et al. [17, 22] proposed a system whereby the base station in the sensor network was used to authenticate the sender of data packets. However, this model makes the assumption that the base station operates under perfect conditions and can detect anomalous nodes or nodes acting maliciously. This is done by storing statistics of node activity. The model also implemented security mechanisms at the packet level where each data packet is encrypted with shared keys to ensure data integrity and source authentication.

There are assumptions made in these protocols that have been replicated in the proposal of ASLAN in this paper. It includes the assumption that the base station is always dependable and that all data stored in a sensor node's memory is secure.

4. Architecture

This section describes the proposal of ASLAN. The proposed architecture incorporates the following:

1. Network topology organisation
    a. Formation
    b. Inserting additional nodes into the network
2. Key management
3. Identifying and isolating aberrant nodes
4. Secure routing

4.1 Network topology organisation

ASLAN supports a network topology that is organised into two distinct roles.

4.1.1 Formation

The architecture considers that the network is composed of sensor nodes, cluster leaders and a base station. The base station is the only interface between the sensor network and the outside. Similar to Undercoffer et al. [17, 22], it is assumed to operate under perfect conditions and also have sufficient power and resources to communicate securely with all nodes and outside the network. Before deployment, each sensor node is assigned a unique ID that is recorded in the base station. After deployment, sensor nodes self-organise into clusters by broadcasting their unique IDs and listening for IDs being broadcast by neighbouring nodes. Upon receiving a broadcast ID, each node adds this ID to its routing table. Nodes that share IDs with each other then form a cluster. Each cluster then elects one sensor node to act as cluster leader and all communication between different clusters must be routed through the respective cluster leader. Similarly, all communication between a node and the base station must also pass through the node's cluster leader.

Since the volume of communication routed through the cluster leader will be significantly larger than that of other sensor nodes in the network, this will increase the cluster leader's power consumption. However, a sensor node's energy supply is very limited so in order to enable consistent power consumption between all nodes in a cluster, the role of cluster leader changes periodically. This provides each node the opportunity of becoming cluster leader.

In this model, when a cluster leader cannot route data accrued by one of its sensor nodes directly to the base station, it may do so by inter-cluster communication and reaching the base station by routing the data via other cluster leaders.

Once the network is deployed, the base station builds a table containing the unique IDs of all the nodes in the network. After the self-organizing process has completed, the base station will then know the topology of the sensor network. Using this hierarchical topology, nodes will collect data, pass this to their respective cluster leader who will aggregate the packets and send them either directly to the base station or via one or more cluster leaders.

4.1.2 Inserting additional nodes into the network

Additional nodes may be inserted into the network at any time. Before a node is inserted, the base station records and stores its unique ID and will insert the node into a cluster having the least number of nodes. This will help minimise the event of a cluster monopolising bandwidth if it contains a greater number of nodes than other clusters who are communicating. The node will then self organise within its cluster.

4.2 Key management

Establishing secure key management in sensor networks is a difficult issue to solve. However, security techniques such as asymmetric cryptography that use keys are impractical due to the sensor node's resource constraints and the network's ad hoc environment where nodes are randomly joining and leaving. One common key management technique employed in wireless sensor networks is a key pre-distribution scheme where key information is embedded in sensor nodes before they are deployed. This is an energy efficient key management mechanism for resource constrained nodes [23].

The key management scheme in this architecture uses two keys similar to that proposed in [23]:

- K_n (network key) – Generated by the base station, pre-deployed in each sensor node, and shared by the entire sensor network. Nodes use this key to encrypt the data and pass onto the next hop.
- K_s (sensor key) – Generated by the base station, pre-deployed in each sensor node, and shared by the entire sensor network. The base station uses this key to decrypt and process the data and the cluster leader uses this key to decrypt the data and extract nonce values.

The base station uses K_n to encrypt and forward data. When a sensor node receives the message, it decrypts it by using its own K_s.

A cluster leader amasses any messages received from nodes within its cluster and forwards them to the next level cluster leader or directly to the base station itself if it is one-hop away. If a cluster leader receives a data packet from a node within its cluster, it will first add its own unique ID and TimeStamp to the packet before forwarding it. All cluster leaders who are not one-hop away to the base station add their own ID to packets they receive from a sending cluster leader.

When the base station receives a packet, it checks the ID of the sending cluster leader. It authenticates the cluster leader who sent the packet and also the packet's integrity.

4.3 Identifying and isolating aberrant nodes

Sensor nodes that do not function as specified must be identified and isolated in order to continue the desired operation of the sensor network. An aberrant node may be the result of an attack or may act maliciously due to unexpected network behaviour. According to Hu et al. [24], an aberrant node is one that is not functioning as specified and may cease to function as expected for the following reasons [17]:

- It has exhausted its power source.
- It is damaged by an attacker.
- It is dependant upon an intermediate node and is being deliberately blocked because the intermediate node has been compromised.
- An intermediate node has been compromised and is corrupting the communication by modifying data before forwarding it.
- A node has been compromised and communicates fictitious information to the base station.

Therefore, the security of the sensor network can be maintained by identifying an aberrant node quickly and isolating it from the sensor network. ASLAN includes a protocol that is used to identify and isolate aberrant nodes. This is divided into two algorithms:

a. node-to-node
b. cluster leader-to-node

In order to describe the functionality of the protocol, it will be assumed that node A wishes to communicate with node B whom are both located within the same cluster. The protocol also assumes that a secure, end-to-end communications channel between node A and node B has been established. It is also assumed that an attacker is not capable of accessing the contents of packets received by the attacked node.

a. node-to-node

Node A will send data (i.e. packets) to node B. Before node A sends a packet, it generates a nonce, appends it to the packet and saves a copy of it in memory. A different nonce is generated for each packet. Due to memory constraints in sensor nodes and the possible large number of nonce values that may need to be generated, the nonce value will be a combination of a random, medium-size prime number and a time stamp. Node A also sends a copy of the nonce value associated with the packet to the cluster leader.

When node B receives a packet, it will be required to send an acknowledgement (ACK) back to node A within a specified time period. This ACK must contain the same nonce that it received. Node B also sends a copy of this nonce value to the cluster leader. Since the protocol assumes that an attacker cannot access the contents of received packets, the attacker cannot access the nonce and therefore append it to the ACK. Therefore, only a genuine node that has not been attacked is capable of sending an ACK containing the correct nonce back to the original sender of the packet.

When node A receives the ACK from node B, it will compare the nonce it receives with that it has saved in memory. If they are the same, this verifies that node B is not an aberrant node. Otherwise, if they are different or if no ACK has been received within the specified time period, it will assume node B is aberrant and node A then sends an alert to the cluster leader. Node A terminates all communication with node B and deletes the nonce value saved in memory.

Likewise, if node A receives an alert from the cluster leader indicating that node B is an aberrant node before receiving the ACK, it will immediately terminate communication with node B and delete all nonce values saved with respect to node B.

b. cluster leader-to-node

When node A sends packets to node B, node A will send the cluster leader a copy of each nonce value for each packet. When node B sends an ACK back to node A containing the nonce value, it also sends a copy of the nonce to the cluster leader. The cluster leader will compare the two nonce values. If they are the same, it will verify that node B has not been compromised and deletes the nonce values saved in memory it received from node A and node B that correspond to the packet.

If the two nonce values are different, the cluster leader issues an alert to all nodes in the cluster that node B is an aberrant node and should be ignored. This alert is also issued to cluster leaders in all other clusters who in turn notify the nodes in their respective cluster. The base station is also alerted and can take measures to isolate or remove node B from the sensor network. Similarly, if the cluster leader receives an alert from node A about node B, it carries out the same procedures.

In a situation when the cluster leader is the sender or receiver of data with another node, then it cannot act as the independent party to receive nonce values and compare them to check for differences. This means that its role of cluster leader must pass to another node that is not currently involved in direct communication. This ensures that the role of cluster leader does change periodically and is shared between all nodes in the cluster.

Table 2. Notation

Timer	*timer*
Node A	*node_a*
Node B	*node_b*
Sent packet	*sent_packet#*
Received packet	*recd_packet#*
Cluster Leader	*cluster_leader*
Base Station	*base_st*
Authentic node	*auth_node*
Aberrant node	*abb_node*

Algorithm 1. Node-to-node

```
1. node_a sends packet to node_b

2. node_a saves sent_packet# sent to node_b

3. node_a sends sent_packet# to cluster_leader

4. if timer not expire then
       node_b send ACK to node_a with recd_packet#
   end if

5. node_a receives ACK from node_b
   if recd_packet# B = sent_packet# A then

       delete sent_packet# in node_a
       node_b = auth_node AND communication continue
   end if

6. if recd_packet# B NOT = sent_packet# A then

       send ALERT to cluster_leader AND
       terminate communication with node_b
   end if
```

Algorithm 2. Cluster leader-to-node

```
1. if sent_packet# node_a = recd_packet# node_b then

       node_b = auth_node
       delete sent_packet# node_a AND recd_packet# node_b
   end if


2. if sent_packet# node_a NOT = recd_packet# node_b OR ALERT received from node_a then

       node_b = comp_node AND send ALERT to
       all nodes in cluster that node_b = abb_node
   end if


3. if sent_packet# node_a NOT = recd_packet# node_b then

       send ALERT to cluster_leader in all clusters
       AND base_st that node_b = abb_node
   end if
```

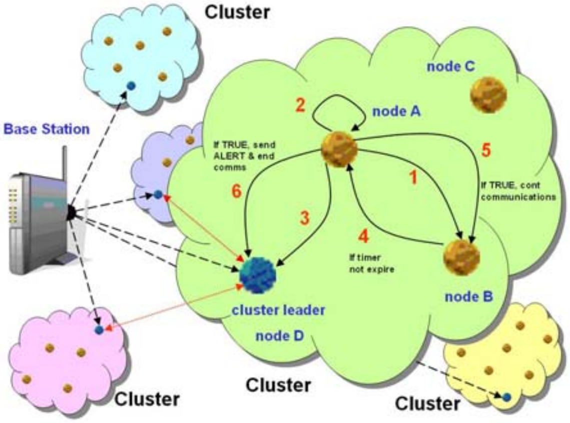

Figure 1. Node-to-node

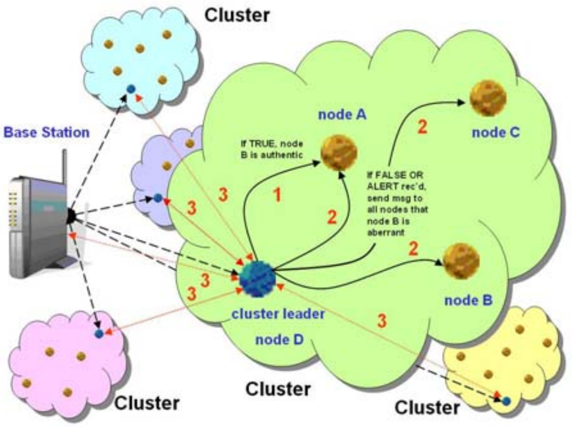
Cluster
Base Station
node A
node C
If TRUE, node B is authentic
If FALSE OR ALERT rec'd, send msg to all nodes that node B is aberrant
1
2
2
2
3
3
3
3
3
3
node B
cluster leader
node D
Cluster
Cluster
Cluster

Figure 2. Cluster leader-to-node

The algorithm presented takes into consideration the nodes and cluster leaders that are not the sender or intended recipient of data or are involved in aggregating nonce values. These nodes forward the data packets without applying any further cryptographic operation, thus further saving the nodes' processing power and memory.

4.4 Secure routing

ASLAN achieves secure data transmission by complimenting the energy efficient secure data transmission algorithm in [25] and adding extra security mechanisms by integrating the proposed algorithm to identify and isolate aberrant nodes in a cluster. The following two algorithms are proposed to achieve secure data communications from node to base station and vice versa.

Sensor node algorithm

1. Node A wishes to send data to another node. The recipient node may/may not exist within the same cluster.

2. Node A generates a nonce value and saves this value temporarily in memory

3. Node A encrypts the data it is sending using the encryption key K_n (assigned at its deployment) and appends its ID, the current TimeStamp, and the nonce value to the encrypted data.

4. Node A sends the encrypted data packet to the cluster leader.

5. Cluster leader receives the encrypted data and makes a copy. It adds its own ID and TimeStamp to the original data packet and forwards this packet to the next higher-level cluster leader or directly to the base station itself if it is one-hop away.

6. Cluster leader decrypts the copy of the data packet it made using it's key K_s (assigned at its deployment) and extracts the nonce value. It stores this temporarily in memory. It discards the copy of the data.

7. If the cluster leader receives incoming data destined for a node within its cluster, then make copy. The cluster leader decrypts the copy using it's K_s and checks if it is an ACK (contains a nonce value). If the data is an ACK, then proceed to step #8, otherwise step #9.

8. Cluster leader compares the ACK nonce value with the original nonce value stored in memory for the original data packet sent. If equal, then delete the nonce value stored in memory and proceed to step #9. Otherwise, send alert to cluster leader that the sender node (node $ID_{1..n}$) to be considered compromised. Discard the original data packet, the copy and delete the nonce value stored in memory.

9. Node A receives data forwarded to it by its cluster leader. Decrypt the data using K_s and check if it is an ACK (containing a nonce value). If not, proceed to step #10. Otherwise, compare the ACK with the original nonce value stored in memory. If same, continue as normal and delete the nonce value in memory. If different, assume sending node is compromised and send alert to cluster leader. Delete nonce value in memory.

10. Reply to sending node. Create a new data packet containing the ACK value (nonce value) extracted in step #9 and encrypt the data using it's encryption key $K_{n.}$ Send the ACK data packet. Process the received data accordingly. Return to step #1.

Base station algorithm

1. If a data packet has been received from a cluster leader that is needed to be forwarded, encrypt it using K_n.

2 If no data packet is needed to be forwarded, check if any incoming data from any cluster leaders. If not, return to step #1.

3. If there is incoming data to the base station, then decrypt the data using K_s. Extract the node ID and the TimeStamp.

4. If the data does not decrypt correctly, discard the packet and proceed to step #6.

5. Extract the message from the decrypted packet and process accordingly.

6. If necessary, send a request to the sensor node that transmitted the original packet to retransmit the data. Return to step #1.

5. Future work

ASLAN is a proposed architecture which assumes that the base station is always dependable and that the data stored in a sensor node's memory is secure. Therefore, the architecture is designed to address situations where an attacker will breach security and cause disruption to the sensor network such as interrupt, intercept, modify or fabricate data packets. Examples of these types of attack include attacks on information in transit [26] and blackhole/sinkhole attacks [27].

Consequently, we will implement and evaluate our architecture with a focus on the following criteria:

- Accuracy of intruder detection
  - Success rate
  - False-positive rate
  - False-negative rate
- Communication overhead (cost)
- Energy consumption

Results for these criteria will then be collated and evaluated.

6. Conclusion

Security is a primary concern in the design of a wireless sensor network. ASLAN needs to be as lightweight as possible in order to reduce the overhead burden placed on sensor nodes that have very limited resources. In this paper, we presented ASLAN: a lightweight architecture that aims to secure a wireless sensor network against deliberate and hostile attack. The proposed architecture consists of phases that involve a model for a self-organising network topology, a secure key management scheme and a secure routing system allowing data to traverse the network securely. The secure key management scheme is based on the pre-deployment of keys to sensor nodes.

ASLAN also incorporates a protocol for the identification and isolation of aberrant nodes. This protocol consists of two sections: node-to-node and cluster leader-to-node, both of whom work in tandem with each other. This protocol will identify any node that has become compromised and isolates it. ASLAN has also been designed so that nodes and cluster leaders, who are involved in forwarding data packets, do not apply any further cryptographic operations and thereby aid in enabling the architecture to be as lightweight as possible.

7. Acknowledgements

The work described in this paper was supported in part by a grant "Secure and Predictable Pervasive Computing" from Science Foundation Ireland (SFI).

8. References

[1] M. Collins, S. Dobson, and P. Nixon (2008), "A Secure Lightweight Architecture for Wireless Sensor Networks". Proceedings of the Second International Conference on Mobile Ubiquitous Computing, Systems, Services and Technologies (UBICOMM 2008), Valencia, Spain

[2] J. M. Kahn, R. H. Katz, and K. S. J. Pister (1999), "Mobile networking for smart dust". In ACM/IEEE International Conference on Mobile Computing and Networking (Mobicom 99), Seattle, WA, USA.

[3] Matt Blaze, Joan Feigenbaum, and Angelos D. Keromytis "Keynote: Trust Management for Public-Key Infrastructures", SpringerLink, Lecture Notes in Computer Science, Security Protocols book, Volume 1550/1999, p625

[4] Al-Sakib Khan Pathan, Hyung-Woo Lee, and Choong Seon Hong (2006), "Security in Wireless Sensor Networks: Issues and Challenges", Proceedings of the 8th IEEE International Conference on Advanced Communication Technology (ICACT), Vol II, pp 1043 – 1048.

[5] D. Boyle, and T. Newe (2008), "Securing Wireless Sensor Networks: Security Architectures". Proceedings of the Journal of Networks (JNW), Vol. 3, No. 1, ISSN : 1796-2056

[6] Blackert, W.J., Gregg, D.M., Castner, A.K., Kyle, E.M., Hom, R.L., and Jokerst, R.M. (2003), "Analyzing interaction between distributed denial of service attacks and mitigation technologies", Proc. DARPA Information Survivability Conference and Exposition, Volume 1, pp. 26 – 36.

[7] Wang, B-T. and Schulzrinne, H. (2004), "An IP traceback mechanism for reflective DoS attacks", Canadian Conference on Electrical and Computer Engineering, Volume 2, 2-5 May 2004, pp. 901 – 904.

[8] Douceur, J. (2002), "The Sybil Attack", 1st International Workshop on Peer-to-Peer Systems.

[9] Newsome, J., Shi, E., Song, D, and Perrig, A (2004), "The sybil attack in sensor networks: analysis & defenses",

Proc. of the third international symposium on Information processing in sensor networks, ACM, 2004, pp. 259 – 268.

[10] Chris Karlof and David Wagner (2003), "Secure routing in wireless sensor networks: attacks and countermeasures", ScienceDirect, Ad Hoc Networks, Volume 1, Issues 2-3, pp. 293 – 315.

[11] Yih-Chun Hu, Adrian Perrig, and David B. Johnson (2002), "Wormhole detection in wireless ad hoc networks," Tech. Rep. TR01-384, Department of Computer Science, Rice University, June 2002.

[12] Pfleeger, C. P. and Pfleeger, S. L. (2003), "Security in Computing", 3rd edition, Prentice Hall 2003.

[13] Mike Chen, Weidong Cui, Victor Wen, and Alec Woo (2000), "Security and Deployment Issues in a Sensor Network", http://www.cs.berkeley.edu/wdc/classes/cs294-1-report.pdf

[14] Adrian Perrig, Robert Szewczyk, J. D. Tygar , Victor Wen, and David E. Culler (2001), "SPINS: Security Protocols for Sensor Networks", Proceedings of the Seventh Annual International Conference on Mobile Computing and Networks, MOBICOM 2001

[15] Adrian Perrig, Robert Szewczyk, J. D. Tygar , Victor Wen, and David E. Culler (2002), "SPINS: Security Protocols for Sensor Networks", Wireless Networks, Vol. 8, 521-534

[16] Tanveer A. Zia (2006), "An overview of Wireless Sensor Networks and their Security Issues", LNCS – Secure Data Management in Reactive Sensor Networks, SpringerLink, Volume 4332

[17] J. Undercoffer, S. Avancha, A. Joshi, and J. Pinkston (2002), "Security for Sensor Networks", CADIP Research Symposium

[18] L. Eschenauer and V. Gligor (2002), "A Key-management Scheme for Distributed Sensor Networks", Proceedings of the 9th ACM conference on Computer and Communication Security", Washington DC, USA

[19] H. Chan, A. Perrig, and D. Song (2003), "Random Key Predistribution Schemes for Sensor Networks". Proceedings of the IEEE Symposium on Security and Privacy, Oakland, California, USA

[20] W. Du, J. Deng, Y. S. Han, and P. K. Varshney (2003), "A Pairwise Key Pre-Distribution Scheme for Wireless Sensor Networks", ACM CCS

[21] W. Du, J. Deng, Y. S. Han, S. Chen, and P. K. Varshney (2004), "A Key Management Scheme for Wireless Sensor Networks Using Deployment Knowledge", IEEE InfoCom

[22] Sasikanth Avancha, Jeffery Undercoffer, Anupam Joshi, and John Pinkston (2003) "Secure Sensor Networks for Perimeter Protection," Computer Networks: The International Journal of Computer and Telecommunications Networking, 43(4), 421–435

[23] Tanveer Zia and Albert Zomaya (2006), "A Security Framework for Wireless Sensor Networks", Proceedings of the IEEE Sensors Applications Symposium, February, 2006

[24] F. Hu, J. Ziobro, J. Tillett, and N. Sharma, "Secure Wireless Sensor Networks: Problems and Solutions", Rochester Institute of Technology, Rochester, New York, USA

[25] H. Cam, S. Özdemir, D. Muthuavinashiappan, and P. Nair (2003), "Energy Efficient Security Protocol for Wireless Sensor Networks", IEEE

[26] Al-Sakib Khan Pathan, Hyung-Woo Lee, and Choong Seon Hong (2006), "Security in Wireless Sensor Networks: Issues and Challenges", Proceedings of the 8th IEEE International Conference on Advanced Communication Technology (ICACT), Vol II, pp 1043 – 1048

[27] Edith C.H. Ngai, Jiangchuan Liu, and Michael R. Lyu (2007), "On the Intruder Detection for Sinkhole Attack in Wireless Sensor Networks" Computer Communications, Vol 30, Issue 11 – 12, pp 2353 – 2364

Michael Collins (B.Sc., M.Sc.) is a part-time Ph.D candidate and member of the Systems Research Group in University College Dublin, Ireland. His research is under the joint supervision of Dr. Simon Dobson and Prof. Paddy Nixon and is focusing on security and authentication in wireless sensor networks. His research interests include ubiquitous computing, security, and I.T. education.

Simon Dobson (B.Sc., M.A., D.Phil) is a lecturer in the School of Computer Science and Informatics, University College Dublin, Dublin, Ireland. His research centres around adaptive pervasive computing and novel programming techniques. He has an extensive record of published work (including papers in CACM, TAAS, JPDC, EHCI and ECOOP) and primary authorship on grants worth over €3M (and further involvements grants worth over €28M) feeding around €1.5M directly into his own current research programme. He is National Director for the European Research Consortium for Informatics and Mathematics, a board member of the Autonomic Communication Forum (at which he chairs the semantics working group), and a member of the IBEC/ICT Ireland standing committee on academic/industrial research and development.

Paddy Nixon (B.Sc., M.A., Ph.D) is Professor of Distributed Systems in the School of Computer Science and Informatics, University College Dublin, Dublin, Ireland. He is the lead of the Systems Research Group located in the UCD Complex and Adaptive Systems Laboratory (CASL). Prof. Nixon is also a SFI (Science Foundation of Ireland) Professor of Ubiquitous Systems. His funding of EUR €2.5M from the SFI is helping to seed the formation of the SRG with an initial core activity in Secure and Predictable Pervasive Systems. He has published numerous articles and proceedings in the areas of pervasive computing, adaptive information, autonomic computing, middleware and applied formal methods.