

A Survey of Wireless Sensor Network Security

December 6, 2004

James Frye Jr.

## Table of Contents

Introduction.....	3
Security Techniques and Protocols.....	4
LiSP: A Lightweight Security Protocol for Wireless Sensor Networks.....	4
TinySec:A Link Layer Security Architecture for Wireless Sensor Networks.....	5
SPINS:Security Protocols for Sensor Networks.....	6
Fast Authenticated Key Establishment Protocols for Self-Organizing Sensor Networks...	7
An Authentication Framework for Hierarchical Ad Hoc Sensor Networks.....	8
A Pairwise Key Pre-distribution Scheme for Wireless Sensor Networks.....	9
Conclusion.....	10
References.....	11

# 1. Introduction

The area of wireless sensor networks is a growing field that has many applications that are being used today and others that are sure to be discovered. Currently wireless sensor networks are being utilized for applications such as habitat monitoring, and military operations. The use of these networks could prove to be critical in the future with the threat of terrorism and war abroad. The homeland security application of a wireless sensor network would give someone the ability to collect and analyze data remotely and detect any kind of biological attack. If the military were to use wireless sensor networks to collect such sensitive data the information passed over the network would have to be secure. This is one aspect of wireless sensor networking that has yet to find a standard that everyone follows. This is due primarily to the structure of the wireless sensors and the networks they create. Wireless sensor networks are composed primarily of an ad hoc network of wireless sensors that collect data and send the information collected back to the base station. Due to wireless sensors having to be micro sized the sensors are severely constrained in memory, processing power, and battery power. Obviously this makes it painstakingly difficult to come up with an algorithm that can be implemented using the memory available and not consuming all of the battery power. There have been several attempts at securing wireless sensor networks which all have their pros and cons. In this paper I am going to look at several techniques that have been developed to secure data over wireless sensor networks. After I have compiled all of the security schemes I will give the outlines for a security scheme of my own that I feel would be best effective.

## 2. Security Techniques and Protocols

### .1 LiSP: A Lightweight Security Protocol for Wireless Sensor Networks

“LiSP: A Lightweight Security Protocol for Wireless Sensor Networks” which was developed Taejoon Park and Kang G. Shin is a lightweight security protocol that was developed as a security protocol with a tradeoff between security and resource consumption via efficient rekeying.[1] The goal of this protocol was to have a rekeying protocol that had 1.) efficient key broadcasting without retransmission/ACKs, 2.)implicit authentication for new keys without incurring additional overhead, 3.)seamless key refreshment without disrupting ongoing data transmission and 4.)robustness to inter node clock skews among nodes.[1].

The architecture of LiSP is based primarily on the use of symmetric keys. The two keys that are used are the master key(MK) and the temporal key(TK). The temporal key is used to encrypt and decrypt data packets, and the master key is used to send a temporal key to a single node. In the example used in the paper, the network nodes were first segregated into groups. After the network has been set up LiSP automatically selects one of the group heads to be a key server(KS). The job of the key server is to distribute the temporal key, authenticate nodes that are new to the network and detect nodes that have been compromised. When a key server transmits a packet for the first time it contains the length of the TK buffer, the key refresh rate, and the initial TK. After the refresh rate has passed, a future TK will be transmitted to the rest of the network as a control packet. The need for a Message Authentication Code is eliminated because the nodes are able to implicitly authenticate the TK by checking to see if the new TK matches the sequence of the other TKs in the TK buffer. When a message is encrypted to be sent within the group, the temporal key is used along with a keystream. The keystream is generated using the TK, nodeID, and the initialization vector(IV). The keystream is then XORed with the plaintext to create the ciphertext. The MAC is generated using the keyed, nodeID, IV, and the plaintext. The packet is then transmitted to the destination node. The process is done in reverse to decrypt the message.

LiSP provides a great deal of protection from compromised nodes and key servers. The keying system with implicit authentication allows the sensor to quickly detect whether or not the key that was sent from the key server is authentic or not. As long as the refresh rate is not very fast the sensors will not run out of battery power at a fast rate. LiSP is very scalable because the key server does most of the calculations and the key server can change depending on whether the key server has been compromised or not.

## **.2 TinySec:A Link Layer Security Architecture for Wireless Sensor Networks**

The “TinySec Link Layer Security Architecture for Wireless Sensor Networks” was designed by using other security protocols as a guide. TinySec was developed by Chris Karlof, Naveen Sastry, and David Wagner at UC Berkeley. The design goal of this project was to create a link layer security protocol that contains access control, message integrity, and message confidentiality. TinySec supports two different security options: authenticated encryption (TinySec-AE) and authentication only (TinySec-Auth)[2]. When authenticated encryption is used the payload is encrypted and the entire packet is authenticated using a MAC. When the authentication only option is used the payload is not encrypted and the packet is authenticated using a MAC.

For encrypting messages the authors chose cipher block chaining(CBC) over stream ciphers. The CBC encrypted messages using a unique Initialization vector(IV) for every time a plaintext was encrypted. This was mainly due to it being possible to figure out the plaintext that is being encrypted if in a stream cipher the IV is ever used to encrypt more than one message. If CBC is used then only the length of the cipher text is leaked. Using CBC and a 4 byte MAC someone trying to inject a malicious packet into the network would succeed after  $2^{31}$  tries.

Unlike LiSP, TinySec is not tied down to any specific keying mechanism. This is because the keying mechanism that is selected will depend on the type of application the wireless sensor network will be used for. This gives the user extra flexibility by giving them the choice of any keying scheme they would like. When TinySec is compiled to a wireless sensor a key file is automatically generated and maintained on the developers computer. Whenever TinySec is compiled again a key is selected from the key file on the pc. TinySec is also cipher independent which allows the user to use whatever cipher they would like.

The drawback for implementing TinySec is that TinySec packets are one to five bytes longer than normal WSN packets. This causes more processor and battery power to be consumed. The cryptography causes more processor power to be consumed and the sensors radio has to be turned on for a longer length of time which causes the battery power to decrease at a faster rate.

### .3 SPINS:Security Protocols for Sensor Networks

The “SPINS:Security Protocols for Sensor Networks” was developed by Adrian Perrig, Robert Szewczyk, J.D. Tygar, Victor Wen, and David E. Culler at UC Berkeley. Spins was primarily based on  $\mu$ TESLA(the micro version of TESLA) and SNEP(Secure Network Encryption Protocol). These security schemes were chosen because “SNEP provides data confidentiality, two-party authentication, integrity and freshness” and “ $\mu$ TESLA provides authentication for data broadcast.” Both of these security schemes are used with a shared secret key to send packets between nodes. The implementation of SNEP adds a random bit string to all the messages that are transmitted. This allows for semantic security because even if the eavesdropper knows the plaintext and cipher text that were encrypted using the same key.  $\mu$ TESLA uses a loosely synchronized timer on both the base station and other nodes to authenticate the MAC key. The phases that  $\mu$ TESLA have are sender setup, broadcasting authenticated packets, bootstrapping a new receiver, and authenticating broadcast packets[3].

A block cipher was used by SNEP to implement all of the symmetric cryptographic primitives such as MAC, and encryption. Out of the block cipher algorithms AES, DES and RC5 the algorithm that was chosen was RC5. RC5 was chosen because of its relatively small code size and it being less computation intensive. A counter is used on both the sending and receiving node in order to encrypt and decrypt messages respectively. If two identical messages are encrypted the resulting packets will look totally unrelated because the packets were encrypted using two different counters.

This protocol was developed by researchers at UC Berkeley so they have a better understanding on what the sensors are capable and not capable of doing. This proves true by them creating a micro version of TESLA in order to accommodate the sensors rather implementing the entire protocol like the project “An Authentication Framework for Hierarchical Ad Hoc Sensor Networks”. Also the synchronized timer is used to authenticate the packets over the network from a single sensor. This makes the sensors less dependent on a third party sensor, keeping the number of packets down to a minimum.

## **.4 Fast Authenticated Key Establishment Protocols for Self-Organizing Sensor Networks**

The paper “Fast Authenticated Key Establishment Protocols for Self-Organizing Sensor Networks” was written by Qiang Huang et. al at Princeton University and Mitsubishi Electric Research Laboratories to create an “efficient authenticated key establishment protocols between a sensor and a security manager in a self organizing network”[4]. In this project the researchers used elliptic curve cryptography(ECC) to provide encryption for the sensor nodes. ECC was chosen because only small key lengths are needed in order to get a reasonable amount of security. To authenticate keys, certificates are used to find out if the public key is in fact a trusted sensor. The certificates themselves are generated by the sensors and security managers before they establish themselves into the network. The computations also can be assisted by a computation server if needed. With the feature of being able to authenticate public keys for each sensor node there is no need to keep a database or buffer of each of the sensors public keys.

In order for someone to find out one of the private keys they would have to figure out the equation:  $z_i = q_i H(\text{MacKey}_i) + d_{ui} \pmod n$ . Cracking this equation is especially difficult because the ephemeral private key  $d_u$  is created through the computation of  $d_u = H(c_u || r) \in [2, n-2]$  with  $c_u$  and  $r$  as random numbers. The person that is trying to find out the private key would literally have to be able to crack multiple equations that contain random values.

Overall the computation for encrypting information takes roughly 760 msec not including hashing, symmetric key operation, key derivation and random number which altogether takes 3 msec. During the execution of the protocol 6 messages were exchanged which had a total of 180 bytes. The 6 messages were “two for mutual authentication and implicit certificates, two for the afterwards link key generation process and another two for the explicit key confirmation.”[4]. This protocol also takes up 5.2K worth of space due to the complexity of the calculations. After finding out how much the energy and processing power the sensors were using with this implementation of the protocol some changes were immediately made. The first change was to replace one of the ECC computations to calculate the link key  $c_u$  the with a Modular Square Root(MSR) technique. The other more complex computations were moved from the sensors to the security manager in order to obtain a processing time of 455 msec on the sensors.

The main drawback of using this key establishment protocol is that sometimes a computation server may be needed for some of the computations. There was no indication on whether the computation server was a node or a computer. The amount of packets that are exchanged to authenticate a key seems like too many to authenticate a key. The strength of this protocol is the difficulty of being able to figure out any of the keys due to the randomness of values.

## **.5 An Authentication Framework for Hierarchical Ad Hoc Sensor Networks**

The project “An Authentication Framework for Hierarchical Ad Hoc Sensor Networks” was developed by Mathias Bohge, and Wade Trappe at the Wireless Information Network Laboratory at Rutgers University. This project addressed the issue of being able to authenticate data that is being passed by sensors of different levels of computational power. In the project sensors were used that were of three levels of computational and communicational power. The difference between a hierarchical sensor network and a regular sensor network are: 1.)varying levels of computational power within the sensor network, 2.)sensors do not communicate with each other, 3.) the forwarding node is a radio relay.[5].

Just like SPINS this project implements a version of TESLA into their authentication scheme. TESLA was chosen due to the lower level sensors not having enough computation power to calculate public keys. Unlike the SPINS project, TESLA is fully implemented and not scaled down from the original version. This is because the certificate authority is a pc. The actual framework of this project is based on a system of trust. When a sensor first enters the network the information that is sent from it initially is not trusted. The sensor has to first send obtain a certificate(iCert) from a trusted third party(TTP) node. When the node or access point wants to join a network it sends its iCert to an application on the computer. If the iCert is valid then the program will then send the secret key to the node or access point in order for it to have access to the network. Nodes and access point are also able to authenticate themselves by obtaining runtime certificates from the application. In order to get rid of compromised nodes certificate renewal is used.

Data is sent over the network in two modes. The two modes are weak mode, and assured mode. In weak mode the node that sends a message to the application does not get information on what access point relayed the message to the application. In assured mode the authenticity of the nodes and access points are proven in order to assure that the data is being passed to the application and not to a compromised node. This is done through sending an assured data request that authenticates and sends a secret key to the nodes and access points.

This project was built upon the work that the SPINS project developed, however this framework was built for a hierarchical ad hoc sensor network. Not all sensor networks are hierarchical therefore this framework may not work on all sensor networks. Also computers are used for the main application and if the main application is compromised then the whole network will go down with it.

## **.6 A Pairwise Key Pre-distribution Scheme for Wireless Sensor Networks**

The paper “A Pairwise Key Pre-distribution scheme for Wireless Sensor Networks” written by Wenliang Du et. al at Syracuse University proposes a key pre-distribution that 1.)substantially improved network resilience against node capture over existing schemes, 2.) Pairwise keys that enable authentication and 3.) Thorough theoretical analysis of security , and communication overhead analysis.[6] The scheme was built upon Blom’s key pre-distribution scheme along with random key pre-distribution. Blom’s key pre distribution scheme has a threshold that allows up to a number of nodes to be compromised before the entire network is compromised. In Blom’s pre-distribution scheme nodes are initially deployed with a set of keys that another node will have in common with it. In order to establish a connection with another node the two nodes must have a key from a common space. If the two nodes have a key from a common space then they can generate a pairwise key that they share. If they do not have a key from a common space then they can create an agreement with another nodes that share a common key.

The keys are generated by creating first 1.)Generating a G matrix, 2.)Generating a D matrix, and then 3.)selecting the spaces for the nodes. After the nodes are deployed to find out if it shares a space with another node they send a packet that contains the node id, the indices of the space it carries and the seed of the column of G it carries.[6]. The local connectivity is calculated to find out if two neighboring nodes have a space in common. After all of the nodes have done this a key sharing graph is created.

This scheme was tested by seeing how many nodes would have to be compromised in order for the entire network to be compromised. To further strengthen the security nodes can establish a random secret key that has nothing to do with any keys in a certain space. This allows for more sensors to be compromised without endangering the rest of the network.

This scheme is a very good scheme in that it allows a great amount of sensors to be compromised without endangering the rest of the network. The idea of sensors finding a common key in a space and then generating a secret key from those values is very hard to crack. The main drawback is that the generation of the matrices may prove to be calculation intensive on the base station.

### 3. Conclusion

In conclusion there are a great deal of security schemes and protocols that are being used and developed for wireless sensor networks. Every one of them surveyed in this paper has their own strengths and weaknesses. I feel as though the best overall security protocol is the TinySec protocol that was developed by researchers at UC Berkeley. I feel as though this protocol leaves enough flexibility to be able to spread across multiple types of wireless sensor networks. The cipher block chaining method along with the random initialization vectors allows for secure transmission without the data being compromised easily. The ability to be able to choose your own keying system is another plus that keeps the users from being tied down to one specific key scheme. The keying scheme I would choose to use with TinySec is the pair wise key distributing scheme.

The pair wise key distribution scheme allows a large number of nodes to be compromised without compromising the entire network. This scheme is extremely powerful in allowing the base station to do the computations of the matrices to distribute a certain column of keys to each node. This allows the major calculations to be done by the computer rather than the nodes having to bear some of the processing. The only things the nodes have to really compute are whether or not the neighboring node has a key in the common space.

The only scheme that I feel as though was not scalable enough is the project called “Fast Authenticated Key Establishment Protocols for Self-Organizing Sensor Networks”. This scheme is a very good scheme however it may be too calculation intensive for sensors that have very limited processing power. The elliptic curve cryptography(ECC) requires multiple calculations that will cause the battery power of the sensor to decrease more rapidly than other schemes.

Overall security for wireless sensor networks is very hard to develop due to the limited resources of the sensors. Unless a majority of the calculations are done by a computer, the sensors are limited in the amount of calculations that are performed. Until the processing power and battery life of wireless sensor networks increase, sensor network security will always be a field in which much work needs to be done.

## 4. References

- [1] T. Park and K. Shin. LiSP: A Lightweight Security Protocol for Wireless Sensor Networks. *ACM Transactions on Embedded Computing Systems*, Vol 3, No. 3, Pages 634-660, August 2004
- [2] C. Karlof, N. Sastry, and D. Wagner. TinySec: A Link Layer Security Architecture for Wireless Sensor Networks. *SenSys '04*. Pages 162-175. November 3-5.
- [3] A. Perrig ET AL. SPINS: Security Protocols for Sensor Networks. *Wireless Networks* vol. 8, 2002, Pages 521-534.
- [4] Q. Huang ET AL. Fast Authenticated Key Establishment Protocols for Self-Organizing Sensor Networks. *WSNA '03*, September 19, 2003, Pages 141-150.
- [5] M. Bohge and W. Trappe. An Authentication Framework for Hierarchical Ad Hoc Sensor Networks. *WiSE '03*, September 19, 2003, Pages 79-87.
- [6] W. Du ET AL. A Pairwise Key Pre-distribution Scheme for Wireless Sensor Networks. *CCS '03*, October 27-31, 2003, Pages 42-51.
- [7] A. Perrig, J. Stankovic, D. Wagner. Security in Wireless Sensor Networks. *Communications of the ACM*, Vol. 47 No. 6, Pages 53-57, June 2004.