Journal of Engineering and Applied Sciences 3 (12): 911-917, 2008
ISSN: 1816-949X

Reliable Communication in Sensor Networks Using KDC

[1]G. Murugaboopathi and [2]V. Khanaa
[1]Bharath University, Chennai, India
[2]Department of Information Technology, Bharath University, Chennai, India

Abstract: In mobile node communications a set of secure protocols that rely on simple network coding operations to provide robust and low-complexity solution for sharing secret keys among sensor node authentication. Key distribution refers to the problem of establishing shared secrets on sensor nodes such that secret symmetric keys for communication privacy, integrity and authenticity can be generated. A key distribution scheme should allow a sensor network to bootstrap (initialize) a secure communications infrastructure providing secrecy (i.e., outsiders cannot derive information from eavesdropping on the wireless communications of the nodes) and preferably entity and message authentication. It should also allow for the addition of new nodes and preferably the revocation of old nodes. In an early publication (SPINs) I proposed a basic symmetric key establishment protocol mediated by a base-station acting as a trusted authority. In my project I use the secret key distribution for mobile node using sensor nodes.

Key words: KDC, QDK, secret key distribution, sensor networks, network coding, cryptography

INTRODUCTION

The potential computational speedup that quantum algorithms offer in certain problems threatens the security of current cryptographic techniques that rely on the infeasibility of factoring large numbers. But the same technology that currently threatens the public-key infrastructure also provides a seeming alternative: a protocol for Quantum Key Distribution (QKD), which provides a secure method for establishing a secret key between 2 participants. These 2 people can then use this key to encrypt information, providing them with the ability to communicate securely. QKD has been implemented numerous times and it's commercially available. Currently, available proposals can be divided into at least 3 basic types of secret key distribution schemes (Du *et al*., 2005). Public-key infrastructure, trusted third party and key predistribution. QKD can be implemented in ways not anticipated until recently. My proposed could use a quantum protocol to obtain a new protocol that's physically indistinguishable from the original but that also contains information channel whose existence is undetectable by any currently known technology. Such hidden channels could potentially provide secure communication without encryption: the protection that Quantum mechanics offers to keys could extend to the information transmitted during communication itself, making key-based encryption unnecessary.

An alternative solution is to use key pre-distribution schemes, such that prior to deployment each node is loaded with a key ring of k keys, chosen randomly from a random pool P, as proposed in Eschenauer and Gligor (2002). A secure link is said to exist between 2 neighboring sensor nodes, if they share a key with which communication may be initiated. A random graph analysis in Eschenauer and Gligor (2002) shows that shared-key connectivity can be achieved almost surely, provided that each sensor node is loaded with 250 keys drawn out of a pool of roughly 100.000 sequences. A different scheme with pre-installed key rings is described in which the network key is erased immediately after the pairwise keys are established. Since, nodes in that situation can no longer establish pairwise keys, the protocol is only suitable for static WSNs.

A hidden channel offers 2 people a way to communicate for example 2 people, Alice and Bob, will engage in a typical instance of Quantum mechanics that, to an outside observer, looks like any other. Alice and Bob say 1 thing but mean another. Like all systems, a quantum system has state. A quantum's system state is represented mathematically by vectors such as (|0> and |1>), or (|->and |+>). These states are examples of what is called qubits. To build a system for secure communication without encryption. To achieve this objective, principle of Quantum Mechanics can be applied. Natural encryption that quantum mechanics offers replaces mathematical

Corresponding Author: G. Murugaboopathi, Bharath University, Chennai, India

encryption. The information they exchange is secure, so in principle, keys aren't necessary. They're communicating securely without encryption. The problem of secret key distribution in a sensor network with multiple scattered sensor nodes and a mobile device that can be used to bootstrap the network. My main contribution is a set of secure protocols that rely on simple network coding operations to provide a robust and low-complexity solution for sharing secret keys among sensor nodes, including pair wise keys, cluster keys, key revocation and mobile node authentication. In spite of its role as a key enabler for this approach, the mobile node only has access to an encrypted version of the keys, providing information-theoretic security with respect to attacks focused on the mobile node. This study include performance evaluation in terms of security metrics and a etailed analysis of resource utilization. The basic scheme was implemented and tested in a real-life sensor network testbed.

This class of network coding protocols to be particularly well suited for highly constrained dynamic systems such as wireless sensor networks. I need Cryptography, suppose Mark wants to send a secret message to his girl friend over an insecure channel; third person could know his secret message. Cryptography divided into 2 sections:

- Symmetric (Public-key) cryptosystem
- Asymmetric (Secret-key) cryptosystem. Some of the example of asymmetrical (public-key) cryptosystem is
- ElGamal cryptosystem
- Elliptic curve cryptosystem
- The merkle-hellman knapsack cryptosystem
- RSA (Ronald rivest, adi shamir, leonard adleman)

The above cryptography system has many disadvantages such as:

- Big prime number factorization is so difficult problem
- Not proven security

Some of the example of symmetrical (public-key) cryptosystem is:

Block type:
- DES
- AES. etc.

Stream type:
- LFSR
- One-time pad. etc.

The cryptography system is today the only provably secure cryptosystem. Even the main problem is difficult to implementation. Quantum Cryptography system, sending a secret key by using the laws of physics to warrant the complete security of the transmission. But the same technology currently threatens public-key infrastructure also provides a seeming alternative: a protocol for Quantum Key Distribution (QKD), which provides a secure method for establishing a secret key between 2 participants. These 2 people then use this key to encrypt information, providing them with the ability to communicate securely.

In study proposed, recent investigations reveal that fundamental quantum components. Such as, QKD can be manipulated in ways not anticipated until recently. Someone can use a quantum protocol to obtain a new protocol that's physically indistinguishable from the original, but that also contains an information channel whose existence is undetectable by any currently known technology. Such hidden channels could potentially provide secure communication without encryption: the protection that quantum mechanics offers to keys could extend to the information transmitted during communication itself, making key-based encryption unnecessary.

A hidden channel offers 2 people a way to communicate what the study consider shortly is a hidden channel within QKD. For example 2 people, Alice and Bob, will engage in a typical instance of QKD that to an outside observer, looks like any other. But when the session is over, they will have secretly communicated. Put succinctly, Alice and Bob say one thing but mean another.

Like all systems, a quantum system has state. A quantum system's state is represented mathematically by vectors such as {|0} and {|1} or {|–} and {|+}. These states are examples of what are often called qubits. Suppose that someone sends us a qubit {|*} that represents either a 0 or a 1. To determine the value of a bit that someone sends us, needs to know in which basis it was prepared. One example of a basis is the X basis, X = {|+, |–} and another is the Z basis, Z = {|0, |1}.

MATERIALS AND METHODS

Requirements for sensor network security (Perrig *et al*., 2002). This study formalizes the security properties required by sensor networks and shows how they are directly applicable in a typical sensor network.

Data conpdentiality: A sensor network should not leak sensor readings to neighboring networks. In many applications (e.g., key distribution) nodes communicate

Table 1: Comparison of secret key distribution schemes

Schemes	Advantages	Disadvantages
Public-key infrastructure (Karlof and Wagner, 2003)	Guarantees authentication and secret key distribution without shared secrets	Heavy requirements in terms of memory, communication, computation and capabilities
Probabilistic key-predistribution	Low computational complexity	Heavy memory requirements and extra communication to identify shared secrets. Links are not secured with probability one
Trusted third party (Eschenauer ans Gligor, 2002)	Low computational complexity	Requires a super node, which can be viewed as a single point of attack
Proposed network coding protocols	Low requirements in terms of memory, processing and communication. Provides means of authenticating the mobile node. Offers extra line of defense. Because mobile node does not know the key. Links are secured with probability one	Requires mobile node for boot-strapping security and tamperresistant sensor nodes

highly sensitive data. The standard approach for keeping sensitive data secret is to encrypt the data with a secret key that only intended receivers possess, hence achieving con_dentiality. Given the observed communication patterns, we set up secure channels between nodes and base stations and later bootstrap other secure channels as necessary.

Data authentication: Message authentication is important for many applications in sensor networks (including administrative tasks such as network reprogramming or controlling sensor node duty cycle). Since, an adversary can easily inject messages, the receiver needs to ensure that data used in any decision-making process originates from a trusted source. Informally, data authentication allows a receiver to verify that the data really was sent by the claimed sender. Informally, data authentication allows a receiver to verify that the data really was sent by the claimed sender. In the two-party communication case, data authentication can be achieved through a purely symmetric mechanism. The sender and the receiver share a secret key to compute a Message Authentication Code (MAC) of all communicated data. When, a message with a correct MAC arrives, the receiver knows that it must have been sent by the sender. This style of authentication cannot be applied to a broadcast setting, without placing much stronger trust assumptions on the network nodes.

If one sender wants to send authentic data to mutually untrusted receivers, using a symmetric MAC is insecure: any one of the receivers knows the MAC key and hence, could impersonate the sender and forge messages toother receivers. Hence, it need an asymmetric mechanism to achieve authenticated broadcast. One of the contributions is to construct authenticated broadcast from symmetric primitives only and introduce asymmetry with delayed key disclosure and one-way function key chains.

Data integrity: In communication, data integrity ensures the receiver that the received data is not altered in transit by an adversary. In SPINS, we achieve data integrity through data authentication, which is a stronger property.

Data freshness: Sensor networks send measurements over time, so it is not enough to guarantee con_dentiality and authentication; we

Quantum Mechanics Operation (QMO)

- Define a basis representation for sender
- Create qubits by comparing the randomly selects input string and with sender basis
- Sent sender generated qubits to receiver
- Receiver received the qubits and compare the qubits with receiver basis representation, from this receiver generates the check bits
- Then receiver send his basis to the sender
- Sender received receiver basis and compare his basis with randomly selects input string, from this sender generates the check bits
- The core concept of this project is sender generate check bits and receiver generated check bits both are same
- Now the sending messages converted to the bits and added this bits with check bits randomly and send to the receiver
- Receiver selects the input string with check bits and extracts the check bits, now he got original string

Mobile Secret Key Distribution (MSKD): A key distribution scheme that exploits the existence of a mobile node, requires only a small number of pre-stored keys and provides a level of security similar to probabilistic private key-sharing schemes, while ensuring that shared-key connectivity is established with probability one. Table 1 shows a comparison of the advantages and disadvantages of my scheme in comparison with various existing alternatives.

A basic key distribution scheme: The key distribution scheme in the case of 2 nodes, suppose that 2 sensor nodes A and B want to establish a secure link via a mobile node S. Although, A and B own different keys that are unknown to S, the latter is capable of providing A and B with enough information for them to recover.

Large-scale secret key distribution: The research describe how this scheme can be used in large-scale sensor networks. Figure 1 highlights the required medications. For simplicity, each global key identifer is assumed to result from the concatenation of the node identifer and the local key identifer (e.g.|n| = 24 bit and |j | = 8 bit). Each sensor node knows both its own identifer n and the local key identifers j (substituting the key identifiers i used in the simplified 2-node scheme presented in the previous study).

The general protocol can be described as follows:

- The sensor nodes perform standard neighborhood discovery by broadcasting their identifiers n and storing in a list L_n the identifiers announced by their neighbors
- S broadcasts HELLO messages that are received by any sensor node within wireless transmission range. Each sensor node sends a reply message containing $\{n, L_n\}$
- When S receives {n (A), Ln (A)} from a node A and {n (B), Ln (B)} from a node B, it checks whethern (A) $\in L_n$(B) and n (B) $\in L_n$(A), n (A) 6 = n (B). If this is the case

Usage of keys: There are several ways to make use of the established pair of keys beyond the traightforward solution in which each node encrypts messages with its own pre-stored key. One possibility is to encrypt the messages using both keys in a double cypher albeit at the cost of non-negligible computational overhead. A less onerous alternative is to form a common session key as a function of the 2 shared keys and use that session key to secure the communication. Session keys can also be generated locally, for example node A generates a random value x, encrypts it using one of the shared keys and sends it to node B, which generates a random value yand sends it back to A, encrypted with the other key).

Naturally, the availability of suitable random number generators is a relevant issue to be taken under consideration. One possible approach to fulfill this gap could be using the pseudo-random generator presented, which is particularly well suited for wireless sensors.

Authentication of mobile node: The single most essential feature of the mobile node is that it can generate all the XOR combinations of pairs of keys. This feature unveils a simple way for a sensor node to check the legitimacy of the mobile node (Fig. 1).

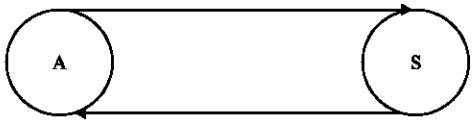


Fig. 1: Sensor node to check the legitimacy of the mobile node

- The sensor node transmits an even number 1 of key identifiers
- The mobile node sends back the key identifiers encrypted by the XOR of the corresponding keys (instead of sending back a message containing the same key identifiers, the mobile node can answer in the challenge-response mechanism with the result of an operation, e.g., the addition of the used local key identifiers)
- The sensor node verifies the authenticity by decrypting and comparing the key identifiers

Similarly, the mobile node and an arbitrary sensor node can agree to encrypt their messages using as key the XOR of the even number of keys indicated by the sensor node, thus ensuring confidentiality.

Request for extra keys: Since, each sensor node is initialized with only a limited number of keys, a situation could occur in which secure links must be established although all the keys have already been depleted.

One way to solve this problem, is for the sensor node to request the mobile node for more keys using the following protocol (Table 2).

- The sensor node transmits an even number of key identifiers and requests a new key
- The mobile node sends back a packet with 2 key identifiers and the XOR of the corresponding keys; the 1st key identifier corresponds to a key already available to the sensor node, whereas the 2nd one refers to a new key; for security this packet is encrypted using the XOR of the even number of keys proposed by the sensor node
- The sensor node decrypts the packet and recovers the new key by performing an XOR operation with the already known key, identified in the received packet

Since, the messages are secure against eavesdropping (even from a legitimate node that shares a key with the requesting node), the latter can securely repeat this process several times up to the total number of admissable local key identifiers. In order to have enough space to save new keys, the requesting node can delete obsolete keys from its memory.

Table 2: Authentication of mobile node

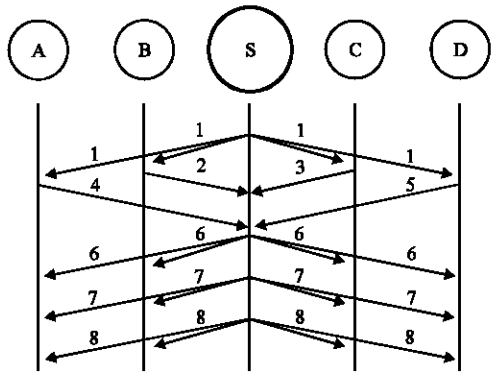
Msg	Sender	Receiver	Content
1	S	A, B, C, D	Hello
2	B	S	{n (B), [n (A), n (C)]}
3	C	S	{n (C), [n (B), n (D)]}
4	S	B, C	{n (B)*j (B), n (C)* j (C), $K_{n(B)*j(B)} \oplus K_{n(C)*j(C)}$}
5	A	S	{n (A), [n (B)]}
6	S	A, B	{n (A)*j (A), n (B)*(j (B) + 1), $K_{n(A)*j(A)} \oplus K_{n(B)*(j(B)+1)}$}
7	D	S	{n (D), [n (C)]}
8	S	C, D	{n (C)*(j (C) +1), n (D)*j (D), $K_{n(C)*(j(C)+1)} \oplus K_{n(D)*j(D)}$}

Cluster keys: Beyond the secret keys that are shared by pairs of nodes, it is often useful to have special keys shared between k nodes in the same cluster. This can be achieved by having nodes exchange their identifiers, build a list of nodes that want to share a cluster key and agree on a common cluster identifier. Once the mobile node learns, which nodes want to share a cluster key, it broadcasts k-1 independent pairwise XOR combinations of keys for each sensor node to be able to recover 1 key for each 1 of the other nodes in the cluster-all it has to do is to solve the resulting linear system of equations. Consider the example shown in Fig. 2, where it can be seen that upon receiving message 6 node A cannot recover any key. However, after receiving message 7, node A can compute the key owned by B and consequently recover the key possessed by C.

Furthermore, upon receiving message 8 node A can also compute the key owned by D. All the knodes proceed similarly until they have k keys in common from which they can compute the cluster key. It can be shown that an eavesdropper will not have enough degrees of freedom to solve the linear system and recover the keys, since, k-1 independent pairwise XOR combinations of k keys are not sufficient to recover any key.

It is fair to say that this approach for generating cluster keys is effective but not necessarily efficient: k-1 transmissions are required from the mobile node for k nodes to be able to communicate with the same cluster key.

Revocation: When a node is captured, it must be possible to revoke its entire key ring. Once the mobile node knows the identifiers of compromised nodes or keys, these can be neutralized in the following manner. First, the mobile node broadcasts a revocation message with the list of node or key identifiers to be revoked. Secondly, the compromised nodes or keys are deleted from the mobile node's memory to prevent the use of exposed keys. Upon receiving the key revocation message, the sensor node verifies if it has any of the revoked keys in its memory. Similarly, upon receiving a node revocation message, a sensor node verifies if it is connected to any of the compromised nodes. After verifying the authenticity of the mobile node the warned sensor node blocks all the connections initiated by the exposed nodes or keys and removes them from its key ring. Revocation affects only the links currently connecting the compromised node to other nodes, thus, isolating it from the network. Chapter 4.



Msg	Sender	Receiver	Content
1	S	A, B, C, D	Hello
2	B	S	{G, n (B), [n (A), n (C), n (D)]}
3	C	S	{G, n (C), [n (A), n (B), n (D)]}
4	A	S	{G, n (A), [n (B), n (C), n (D)]}
5	D	S	{G, n (D), [n (A), n (B), n (C)]}
6	S	A, B, C, D	{G, n (B)*j (B), n (C)*j (C), $K_{n(B)*j(B)} \oplus K_{n(C)*j(C)}$}
7	S	A, B, C, D	{G, n (A)*j (A), n (B)*j (B), $K_{n(A)*j(A)} \oplus K_{n(B)*j(B)}$}
8	S	A, B, C, D	{G, n (C)*j (C), n (D)*j (D), $K_{n(C)*j(C)} \oplus K_{n(D)*j(D)}$}

Fig. 2: Cluster keys

Main algorithm:

- Alice chooses a random string k of roughly 4n bits containing the eventual key
- Alice randomly codes each bit of k in either the X = {|+, |–} or Z = {|0, |1} bases
- Alice sends each resulting qubit to Bob
- Bob receives the 4n qubits, randomly measuring each in either the X or Z basis
- Alice announces in which basis she originally coded each bit of k
- Bob tells Alice which qubits he measured in the correct basis (but not the bit values he obtained); they now share roughly 2n bits
- Alice selects a subset of n bits from the group she formed in step 6 that they will use to check on interference by an eavesdropper (Eve) and tells Bob which bits she selected
- Alice and Bob compare their values of the n check bits; if more than an acceptable number disagree, they abort the protocol (eavesdropping)

- Alice and Bob perform information reconciliation and privacy amplification to select a smaller m-bit key from the remaining n bits

Modules

Mobile server: A server is a computer system that provides services to other computer systems called clients over a computer network. A server is anything that has some resource that can be shared.

The main server is started, in order to make the client to access the data. The main server is connected with the cache server using sockets. The data is transferred between the main server and the cache server through the unique port number.

Mobile client: A client is simply an entity that wants to gain access to a particular server. The client can access the data from cache server by providing the IP address of the server and transfers its data to the server.

Sharing node: It is a sensor node that used to form the secret key by obtain the own keys from the mobile server and the mobile client. Then it sends the XOR key to both the systems. And then it establishes the communication between them.

Networking: The term Computer network refers to the interconnected collection of autonomous computers. Two computers are said to be interconnected if they are able to exchange information. The users must explicitly log onto one machine, explicitly submit jobs remotely, explicitly move files around and generally handle all the network management personally.

A socket is 1 endpoint of a 2-way communication link between 2 programs running on the network. A socket is bound to a port number so that the TCP layer can identify the application that data is destined to be sent.

Security: It gives a Security to the information by using XOR operation of the logic functions. And it uses the DES cryptography algorithm to form the keys to secret key for encryption and decryption function for the server and the client.

Attacker model: This study consider the threat posed by an attacker with the following characteristics:

- The attacker can listen to all the traffic over the wireless medium
- The attacker is able to inject bogus traffic in the network
- The attacker can gain access to the memory of the mobile node or to the memory of a limited number of sensor nodes (but not to both)
- The attacker is computationally bounded (polynomial in the security parameter) and thus, unable to break hard cryptographic primitives

We are now ready to comment on each one of these characteristics. The first characteristic grants that attacker access to all communications. Although this is the strongest possible eavesdropping assumption in a wireless network (Karlof and Wagner, 2003), the threat to our schemes is negligible because the shared keys cannot be decoded from the XOR messages transmitted through the wireless medium. Fake transmissions resulting from characteristic 2, which typically amount to a Bizantine modification attack, can be detected by the legitimate nodes, who may choose to ignore any messages that are corrupted by an invalid key. Characteristic 3 amounts to a physical attack similar to the ones considered in (the implications to our protocol of the threat posed by an attacker who gains access to the memories of both the mobile node and at least one sensor node shall be discussed in Section IV-C). Finally, the limitation on the computational power of the attacker of characteristic 4 is the typical assumption that allows sensor nodes to use standard cryptographic primitives to cipher the sent messages and achieve computational security.

Brute-force attack analysis: A well-known physical argument states that a 256 bit symmetric key is secure against a brute force attack. On the other hand, it is proven in Becher *et al.* (2006) that a brute force attack on a sensor node using a 40 bit key will succeed on the average after 128 years, which is beyond the expected lifetime of current sensor nodes (Becher *et al.*, 2006). We conclude that the result of a brute force attack directly on the sensor nodes is restricted to a Denial of Service attack and if a large enough key is used, an offline attack over captured messages will be useless.

RESULTS AND DISCUSSION

We implemented the basic secret key distribution scheme on a sensor networking testbed, consisting of TelosB motes (UC Berkeley, Crossbow) running the TinyOS 2.0 operating system. Random sequences of 64 bit were pre-distributed on four motes along with the corresponding 8-bit local key identifiers. We also stored in the memory of each mote its own identifier n (•) of 8 bit and a list Ln (•) containing the identifiers of the nodes with which it wants to communicate.

Another mote played the role of the mobile node. In its memory, we stored the keys used by the other 4 motes, in all cases encrypted with a Vernam cipher and included the corresponding identifiers. In the experiment, the mobile node periodically broadcasts its HELLO messages. Upon receiving these HELLO messages, each mote sends back a message with its n (•) and Ln (•). The mobile node, upon receiving this message, verifies if each sensor node whose the identifier is listed in Ln (•) already informed the mobile node that it wants to communicate with n (•). For the nodes that do not satisfy this statement, the mobile node stores the information that n (•) wants to communicate with them. For the other nodes, the mobile node sends back a message containing the identifiers of the 2 nodes (n (•) and the one contained in Ln (•)), the local key identifier that each node has to use and the XOR of the corresponding keys. This message is received by the pair of sensor nodes and each 1 of them recovers the key of the neighboring node by running an XOR of the received data with its own key (corresponding to the received local key identifier). To check if the key was well decrypted by the 2 nodes, each of them sends the neighbor's key in a message that is captured by an observer mote, programmed as a base station and connected to a standard personal computer. May 29, 2008 DRAFT 19 We monitor all the communication steps using the net.tinyos.tools.Listen application. This research shows that the scheme can be easily implemented in real sensor nodes and it works also for large-scale sensor networks.

CONCLUSION

The use of Quantum Computation, momentum and angular momentum of Qubits has been calculated. The use Quantum Mechanics Qubits has been transmitted over a secure channel. This proposed research, operation of Quantum mechanics and secure transmission should be implemented using java, Rmi Programming. It presented a secret key distribution scheme for large sensor networks. This is not a probabilistic scheme, i.e. any 2 nodes that can reach each other can communicate securely with probability 1, using a small number of pre-stored keys albeit at the expense of a mobile node for bootstrapping. It presented several security extensions that exploit network coding to provide secret key distribution in large and dynamic sensor networks. Since, my protocol and its extensions can easily accommodate for additional nodes, new keys and secured links. The proposed network coding approach to be well suited for dynamic sensor networks with stringent memory and processing restrictions.

Thus, conceptual results and practical implementations show that this approach leads to effective ways of generating pairwise and cluster keys, revoking compromised keys, authenticating and defending the mobile node used for bootstrapping the network. It believe that some of the proposed techniques will find natural applications also in other classes of mobile *ad-hoc* networks composed of devices with limited processing and transmission capabilities. Although this use of network coding was so far limited to XOR operations, using linear combinations of symbols is likely to yield more powerful schemes for secret key distribution. Thus, part of this ongoing research is devoted to exploiting random linear network coding and extending these ideas to multi-hop secret key distribution in highly dynamic networks.

REFERENCES

Becher, A., Z. Benenson and M. Dornseif, 2006. Tampering with Motes. In: Clark, J.A., R.F. Paige, F. Polack and P.J. Brooke (Eds.). Real-world Physical Attacks on Wireless Sensor Networks, in Security in Pervasive Computing. Third International Conference, SPC, York, UK, Proceedings, Ser. Lecture Notes in Computer Science, 3934: 104-118.

Du, W., J. Deng, Y.S. Han, P.K. Varshney, J. Katz and A. Khalili, 2005. A pairwise key predistribution scheme for wireless sensor networks. ACM. Trans. Inform. Syst. Secur., 8 (2): 228-258.

Eschenauer, L. and V.D. Gligor, 2002. A key-management scheme for distributed sensor networks, in CCS. Proc. 9th ACM Conf. Computer and Commun. Security. New York, USA: ACM Press, pp: 41-47.

Karlof, C. and D. Wagner, 2003. Secure routing in wireless sensor networks: Attacks and countermeasures. Ad Hoc Networks, 1 (2-3): 293-315.

Perrig, A., R. Szewczyk, J.D. Tygar, V. Wen and D.E. Culler, 2002. SPINS: Security protocols for sensor networks. Wireless Networks, 8 (5): 521-534.