Scalable Session Key Construction Protocol for Wireless Sensor Networks

Bocheng Lai
bclai@ee.ucla.edu

Sungha Kim
yevgeny@ee.ucla.edu

Ingrid Verbauwhede
Ingrid@ee.ucla.edu

Department of Electrical Engineering
University of California, Los Angeles
Los Angeles, CA-90095

Abstract

The security of sensor networks is ever more important nowadays. In this paper we propose a new protocol BROSK to construct link-dependent keys by broadcasting key negotiation messages. The link-dependent key will be negotiated in an ad-hoc scheme. Most of the proposed security protocols in sensor networks are based on point-to-point handshaking procedures to negotiate link-dependent keys, but this will influence the scalability of the network. The simulation shows that the scalability of BROSK is better than two other security protocols of sensor network, SPINS and SNAKE, and is reasonably secure. This new protocol consumes less energy by reducing the number of transmissions used for key negotiation among sensor nodes.

1 Introduction

Distributed sensor networks can be used in a wide range of applications, such as environment monitoring, rescue missions, and smart houses. A lot of interest and effort are being focused on this new network topic. The sensor networks are constructed by a large number of nodes with ultra-low power computation and communication units **[2]**.

Security is an important issue for sensor networks. Stajano and Anderson are the first to point out the battery exhaustion attack **[1]** and this attack can easily be triggered on the sensor nodes with limited energy supply. A more detailed discussion on energy exhaustion attacks will be conducted in section 5.3. We propose a new idea to solve the authentication problem by constructing trust levels among the nodes. Nodes start off at an equal low trust level, and trust between nodes will grow over time after authenticated communication among nodes. A similar idea of distribution of trust is also proposed by Zhou and Hass **[3]**. However how to construct the security among large scale sensor network in an ad-hoc and distributed scheme is still an open question.

Most of the security algorithms are based on sharing a secret key between two parties. They use this shared-secret key to verify each other and even use the key to encrypt and decrypt the data. In a distributed sensor network, constructing and negotiating this secret key is very hard, because of their limited resources. A key server solution has been proposed **[4]**, but that will limit the scalability of the network.

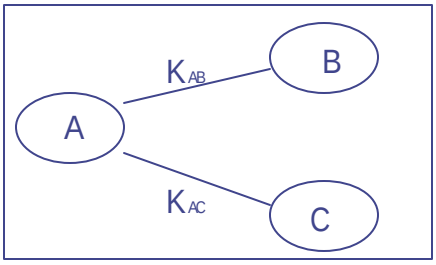

Figure 1 : Link-dependent Key

In this paper, we propose a new protocol to construct this shared-secret key among sensor nodes. The shared-secret key is also known as shared-session key or link-dependent key. The reason is that different links will use different shared-secret keys. For

example, as shown in **Figure 1**, node A has a link to node B and also has a link to node C. If node A wants to talk to node B, it should use key $\mathbf{K_{AB}}$. If node A wants to talk to node C, it should use key $\mathbf{K_{AC}}$. These two keys are independent, therefore even if the adversary can compromise one link key, the rest of the network is still safe. For rest of the paper, we will refer to this as the **shared-session key**.

In section 2, we will briefly introduce two related security protocols for sensor network. We will mainly focus on the protocol to establish the shared-session key among sensor nodes. We will use it as a basis for comparison. In section 3, we will propose a new protocol, BROSK, that can negotiate the key more efficiently. Simulation results will be showed in section 4 and section 5 will evaluate this new protocol and compare the performance and scalability with other protocols.

2 Related Work

This section will briefly introduce two shared-session key negotiation protocols for sensor network, SPINS **[4]** and SNAKE **[5]**. These two protocols have sets of steps to establish the security of sensor network, but here we will mainly focus on the protocol they use to establish the shared-session key among sensor nodes.

2.1 Notation

Following is the convention we used to describe the protocol in this paper.

- **A | B** : data A concatenates with data B
- $\mathbf{\{A\}_{K_{AS}}}$: encryption of data A by key K_{AS}
- $\mathbf{MAC_K[A]}$: MAC (message authentication code) of data A created by key K.
- $\mathbf{N_A}$: the nonce generated by node A. Nonce is a one-time random bit-string, usually used to achieve freshness.
- $\mathbf{ID_A}$: the name of node A.

2.2 SPINS

SPINS is a security suite for sensor networks. It includes two protocols, SNEP and µTESLA. The former is for confidentiality, two-party data authentication, integrity, and freshness and the latter provides authentication for data broadcasting. Here we focus on the key negotiation protocol. As shown in **Figure 2**, assume that node A wants to establish a shared-session key SK_{AB} with node B through a trusted third party S, the central key distribution center (**KDC**). This is a server that can perform authentication and key distribution.

Node A will send a request message to node B (**Figure 2-a**). Node B receives this message and sends a message to the key server (**Figure 2-b**). Key server S will perform the authentication and generate the shared-session key and send the key back to node A and node B respectively (**Figure 2-c and 2-d**). The use of the central key server limits the scalability of the sensor networks.

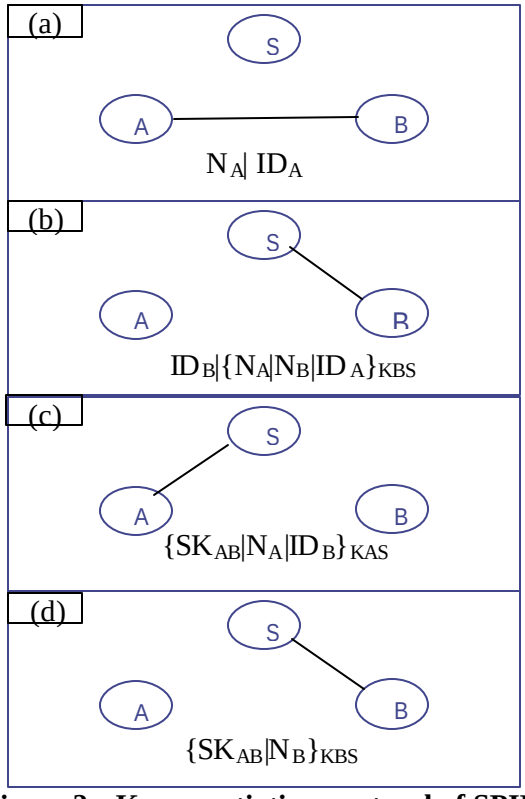

Figure 2 : Key negotiation protocol of SPINS

2.3 SNAKE

SNAKE is a protocol that can negotiate the session key in an ad-hoc scheme. Nodes do not need a key server to perform the key management.

First, node A will send a request to node B (**Figure 3-a**). Node B will reply a message as a challenge to node A (**Figure 3-b**). When node A receives this message, it will prove its authenticity and send the message back to node B (**Figure 3-c**). This is a mutual challenge and authentication procedure. After this three handshaking and mutual authentication procedures, node A and node B will use $\mathbf{K_{AB}}$ as their shared-session key.

$$K_{AB} = MAC_K[N_A|N_B]$$

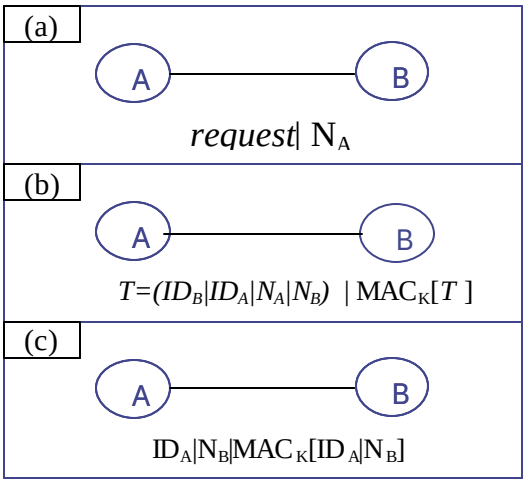

Figure 3 : Key negotiation protocol of SNAKE

3 BROadcast Session Key (BROSK) Negotiation Protocol

BROSK is a new protocol **:** each node can negotiate a session key with its neighbors by broadcasting the key negotiation message. BROSK uses a fully ad-hoc scheme to negotiate the session key and can perform this key negotiating process efficiently. Moreover the scalability of BROSK is significant especially when applied to large scale sensor networks.

3.1 Assumptions

Here we will describe the basic assumptions that we made to construct our protocol.

Assumption 1.
Nodes are resource constrained.
Typical large scale sensor network applications distribute the small sensor nodes in area and power constrained environments. The resource of each node is extremely limited **[2]**.

Assumption 2.
Nodes are static or have a low mobility
Allowing all the nodes to be mobile at the same time will make the problem much more complicated. Actually in many applications, the nodes are fixed in one position for the whole life time. For example, building climate control information is collected by the nodes in the source area, say a cubicle area, and relayed by other nodes to the destination, say the central control room. Of course it is possible that nodes are mobile, this will be related to other network layers, protocols and algorithms. The exploration to all mobile nodes is a topic of further research.

Assumption 3.
Nodes share a master key
Every node in the same network has a shared master key that is never disclosed. This is the key on which the node can tell whether another node is in the same network or not. Also nodes will use this key to authenticate other nodes and negotiate the session key. Of course this master key should be kept in secret. We also assume that the master key will not be extracted from the captured node.

3.2 Broadcast the key negotiation message

A sensor node will try to negotiate a shared-session key by broadcasting the key negotiation message. Each node tries to broadcast the following message: $ID_A|N_A||MAC_K(ID_A|N_A)$

Here ID_A is the name of node A and different

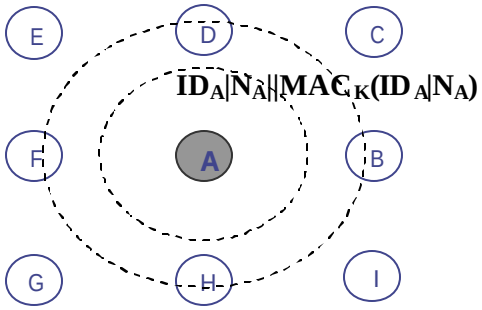

Figure 4: Node A broadcasts the key negotiation message

nodes have different ID. Once a node receives the introducing message broadcasted by its neighbor, it can construct the shared-session key by generating the MAC of two nonces. For example, in **Figure 4**, node B will receive the broadcast message from node A. Node A will also receive the broadcast message from node B (**Figure 5-a**). They can use K_{AB} as their shared-session key (**Figure 5-b**).

$$ID_B|K_B||MAC_K(ID_B|N_B) \quad (a)$$

$$K_{AB}=MAC_K(N_A|N_B) \quad (b)$$

Figure 5:(a) message broadcasted by node B (b)shared session key of node A and node B

3.3 Re-negotiate the key

When the sensor network has been working for a while, nodes might run out of session keys. It is insecure to reuse the same key for data transmission and will be easily compromised. Therefore nodes in sensor network need to re-negotiate new session keys.

4 Simulation Results

Our simulation is conducted on a sensor network simulator developed by NESL**[7]** at UCLA. The simulator uses a c-based discrete-event simulation language PARSEC **[8]**.

We set up our simulation by constructing a grid topology with N by N sensors as shown in **Figure 6**. In the simulator, the sensor nodes use the wireless media in a CSMA (Carrier Sense Multiple Access) scheme and each transmission needs one time slot.

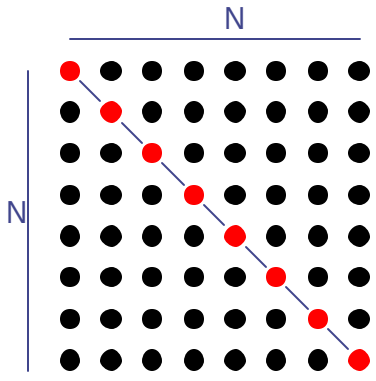

Figure 6: Grid topology

Here we tried two different situations. First is that sensor nodes know nothing about their neighbors, so they just try to broadcast the key negotiation message in a simple distributed random scheme. We ran 20 simulations for different number of nodes and get the average. Each node can only receive signals transmitted by nodes next to it, and these nodes are defined as **neighbors**. How many neighbors does one node have is the **node density**. Collisions between transmissions happen because of simultaneous transmissions in the same time slot and cause the average number of neighbors that one node can really recognize to be only 60% of the actual number of neighbors one node has. We will use **neighbor recognition rate** (NRR) to refer to this rate.

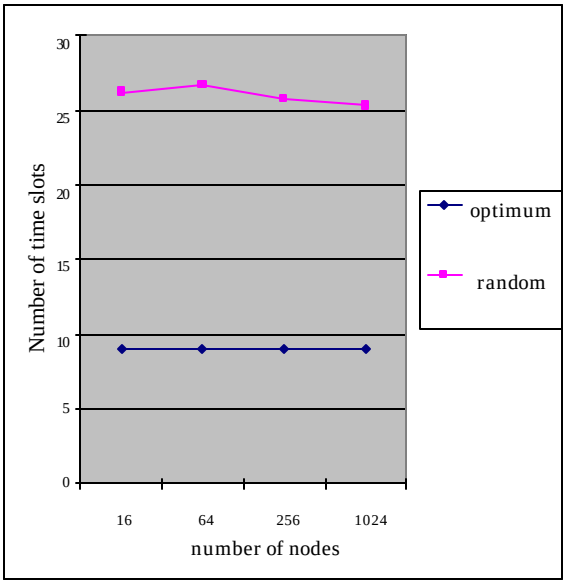

Figure 7 : Number of time slots needed by BROSK to finish key negotiation

The second situation is that sensor nodes know their neighbors and they have already constructed an optimum schedule policy to access the wireless media without causing collisions. **Figure 7** shows the simulation results for BROSK. It shows how many time slots are needed to finish the key negotiation process for different number of nodes. From the simulation result at **Figure 7**, we can realize that this protocol is very scalable. The time

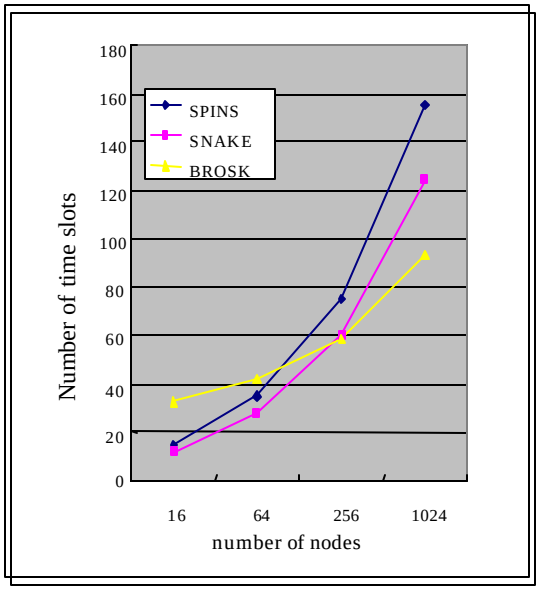

Figure 8 : Number of time slots needed to transmit data when NRR is 60%

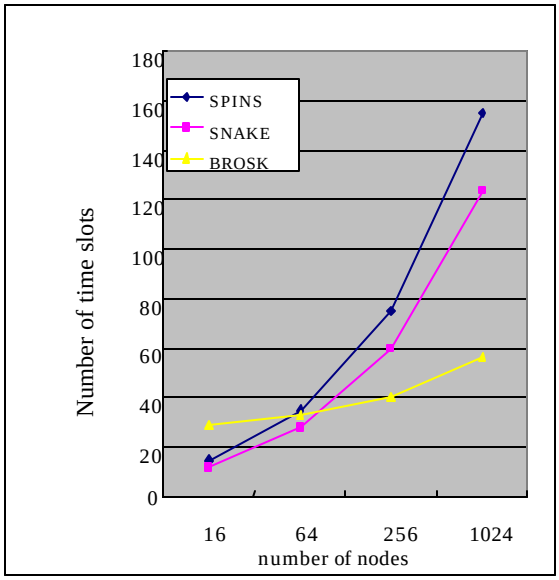

Figure 9: Number of time slots needed to transmit data when NRR is 90%

needed for a sensor network with 16 sensor nodes is close to the time needed for a sensor network with 1,024 sensor nodes. This property remains in both random schedule and optimum schedule.

The next set of simulations show how many time slots are needed to transmit data from the lower-right node to the upper-left node (**Figure 6)**. This simulation includes key negotiation among sensor nodes and transmitting data. **Figure 8** shows the number of time slots needed to transmit data from the lower-right node to upper-left node when NRR is 60%, which means each node can recognize 60% of its neighbors. Here we can see that SPINS and SNAKE perform better than BROSK when the number of sensor nodes is smaller than 64. But BROSK outperforms other two protocols when the number of nodes is large. This is because BROSK needs a certain number of time slots to finish the key negotiation, and this number depends on node density.

Some transmissions fail because of the time slot collisions. But the problem of collision can be mitigated when the sensor network has constructed a good schedule for allocating time slots. The same simulation with NRR is 90% is showed in **Figure 9.** It shows the results of the three protocols. BROSK outperforms the other two protocols when the number of nodes is larger than 64 nodes and improves for larger networks.

5. Evaluation

5.1 Scalability

From the protocol and the simulation results, we can conclude that the BROSK is highly scalable, because the time needed to finish the key negotiation process depends only on the average number of neighbors rather than the total number of nodes. This property holds not only when we use a simple random distributed algorithm but also when nodes have already constructed a good schedule policy to allocate the time slots for key negotiation.

5.2 Power Saving

Transmission of data consumes energy. Therefore the more the transmissions in the network, the more energy will be consumed.

SPINS needs four data transmissions to finish the key negotiation process. In SNAKE, three data transmissions are needed. In BROSK, each node only needs to broadcast once in order to finish the key negotiation process. This situation will be significant when the scale of the network is large, say thousands of nodes.

5.3 Security Analysis

SPINS and SNAKE do not provide a solution for denial of service (DoS) attacks when the malicious node keeps sending the request to negotiate a session key. Both protocols can achieve authentication requirement. But they cannot detect or prevent the DoS attacks, because one adversary can easily trigger a **REPLAY** attack **[9]** and exhaust the energy in the sensor nodes.

In SPINS, the malicious node can simply send the request message for key negotiation continuously, and Node B will keep asking the server about session key with the malicious node. Therefore node B will eventually run out of the energy. However, the base-station may have the ability to detect and try to prevent this attack.

In SNAKE, DoS attacks can be triggered by the same mechanism and SNAKE does not provide the detection of DoS when a malicious node tries to send the message to request key negotiation. In SNAKE there is no base-station to perform attack detection for sensor nodes, every node has to detect this attack by itself and this function is a heavy burden for resource constrained sensor nodes.

However, in BROSK, there is no DoS attack issue when nodes are broadcasting, because each node only broadcasts once and will not response to false request for key negotiation generated by a malicious node. For example, if one malicious node keeps sending the key-negotiation message to its neighbors, nodes in BROSK only need to update the shared-session key to this "malicious node" and do not need to transmit signal like SNAKE or SPINS do. This will eliminate the chance of malicious node to achieve battery exhaustion attack by triggering radio transmission of nodes.

6 Conclusion

In this paper we proposed a new protocol, BROSK, to construct the shared session key in wireless sensor network. BROSK shows great scalability in simulation because the time needed to finish key negotiation does not depend on the number the sensor nodes. We also show that this new protocol can save power by reducing the number of transmissions.

7 Future Work

Our next step is to develop a complete security protocol for sensor network, including authentication, data integrity and confidentiality. Also we are trying to integrate BROSK with other protocols of sensor network in different layers, e.g. media access control layer and network layer.

We already implemented BROSK on 8-bit 8051 processor, which takes under 1 KB including MAC generation. Also we plan to implement BROSK on AVR processor and test the performance on a real platform.

8 Acknowledgement

The authors would like to acknowledge the support of the National Science Foundation research part CCR-0098361. The authors also want to thank the helpful comments from Mani Srivastava and help of NESL group at UCLA.

References

[1] F.Stajano and R.Anderson, "The Resurrecting Duckling: Security Issues for Ad-hoc Wireless Networks,"B.Christianson, B.Crispo and M.Roe (Eds.) Security Protocols, 7th International Workshop proceedings, LNCS,1999.

[2] D.Estrin L.Girod, G.Pottie, M.Srivastava, "Instrumenting the World with Wireless Sensor Networks," IEEE ICASSP 2001, p.2033-2036, vol.4, 2001.

[3] L.Zhou and Z.Hass, " Securing ad hoc network," 1999. IEEE Networks Special Issue on Network Security, Dec, 1999.

[4] Adrian Perrig ,Robert Szewczyk, Victor Wen,David Culler,and J.D.Tygar SPINS: Security Protocols for Sensor Networks, MobiCom, July 2001.

[5] S.Seys, " Key Establishment and Authentication Suite to Counter DoS Attacks in Distributed Sensor Networks," unpublished manuscript, COSIC

[6] W.Diffie and M.E.Hellman. New directions in cryptography. IEEE trans, Inform. Theory, IT-22:644-654, Nov 1976.

[7] Network & Embedded Systems Laboratory (NESL), http://nesl.ee.ucla.edu

[8] Parallel Simulation Environment for Complex Systems (PARSEC), http://pcl.cs.ucla.edu/project/parsec

[9] Bruce Schneier, " Applied Cryptography," Katherine Schowalter publish, 1996.