A-SMOCK: Authenticated SMOCK for Wireless Sensor Networks

Ashly Thomas
PG Scholar
IT Department
Karunya University, India

N. Dhinakaran
Assistant Professor
IT Department
Karunya University, India

P. Anitha Christy Angelin
Assistant Professor
IT Department
Karunya University, India

ABSTRACT
The most major requirements of wireless networks in the current network context are security, scalability and memory efficiency. Most of the earlier implementations use cryptographic key exchanges alone for imposing security but mostly proved to be less secure with the presence of many attacks like IP-spoofing and masquerading. This is because, the security is present only during communication i.e., only message integrity is ensured and not the user authenticity. Hence we are trying to include an authentication scheme along with the already existing integrity scheme to provide maximum security all through the lifetime of networks. In this paper, we consider different options for providing authentication, analyze them and find which method can be used for producing better results

General Terms
Wireless Sensor Networks, Security, Key Management, Authentication.

Keywords
ASMOCK - Authenticated SMOCK.

1. INTRODUCTION
In the current world where the whole world is computerized, with numerous numbers of applications, major of the common people's applications are run in Wireless Sensor Networks and security service is demanded to be available "anywhere" and "anytime". The WSN is built of nodes, from a few to several hundreds or even thousands, where each node is connected to one sensor [1]. Each such sensor network node has typically several parts: a radio transceiver with an internal antenna a microcontroller, an electronic circuit for interfacing with the sensors and an energy source, usually a battery. Designing a key management scheme in current mission-critical networks to fulfill the required attributes of secure communications, such as data integrity, authentication, confidentiality, non-repudiation, and service availability is very challenging. And, to implement security in a communication, the common thing which everyone does is employing cryptographic keys which are effective to an extent. But some attributes of wireless networks like unreliable communication, limited bandwidth, network dynamics with mobile nodes, large number of nodes sharing limited amount of resources etc makes the use of cryptographic keys a challenge. And the main security issues of WSN are small memory, less battery life and less processing capabilities [2]. A certificate based authentication was used [3], [4] but is less used because of the high overhead which also reduces the service availability. Therefore, it is very clear that a self-contained key-management scheme [5] is needed, which allows a mobile node to posses all of the necessary information, the public keys of all other nodes in the network, for authentication locally. One of the very popular concepts of self-contained key management scheme is a trusted centralized server or centralized authenticator with which the nodes can communicate freely before they have been deployed into the network and nodes contact the authenticator for ensuring secure communication between others. The most used security scheme in wireless networks is public key cryptosystem which is found very practical for resource-limited networks[6].

In this paper, we are designing a self-contained public-key-management scheme which in contrast to the traditional schemes, authentication procedure does not require certificate exchanges; instead nodes need to know the ID of the other party in communication to identify the public keys of the other. Also, this scheme uses a smaller set of cryptographic keys and a sender uses multiple keys to encrypt a message and a receiver needs multiple keys to decrypt the message.

This will reduce the number of keys used and thus increase memory efficiency, but the security and confidentiality provided is by the verification the node ID alone, which will be received when a node needs to communicate with another. There can be occurrence of many attacks like, in a condition where an attacker masquerades as a valid node, sending its own ID to the node which requests for communication. When the sender node gets an ID as per its request, it is not easy to find whether ID is sent by the actual node. We considering this problem and adding an authentication with the above mentioned public key management system. Thus this paper proposes a self organized public key management system which is more secure with integrity, confidentiality and memory efficiency.

2. OVERVIEW OF SMOCK
SMOCK is a self-contained public key management scheme [7] which is able to resist the Sybil attack, achieves zero communication overhead for authentication, and offers high service availability. In this scheme, small number of cryptographic keys is stored off-line at individual nodes before they are deployed in the network. To provide good scalability in terms of number of nodes and storage space, combinatorial design [8] of public-private key pairs is used, which makes sure that a set of keys with one user will not be a subset of keys held by another. According to this, a key chain will be allocated to each node in the network and accordingly uses multiple keys to encrypt and decrypt.

The overall scenario of the self-contained key management scheme will be as follows. Before mobile devices are dispatched to an incident area, they are able to communicate securely with the trusted authentication server in their domain center, and get

prepared before their deployment. Each node possesses a unique combination of private keys, and knows all public keys. The private key combination pattern is unambiguously associated with the node ID. It means, if a sender *A* wants to send a message to receiver *B*, *A* will first acquire *B*'s ID to infer a set of private keys owned by *B*. Then *A* will encrypt the message with the public key set that corresponds to the private keys of *B*. This scheme uses the isometric key allocation algorithms to achieve the objectives of a good key management protocol. And, for allocating keys, we need to find the following.

1) For a given network, how to determine *x* and *y*;
2) How to allocate distinct private key sets to users to achieve secure communication between each pair of users,

where 'x' is the number of key pairs used by the network, and 'y' is the number of private keys possessed by each node. The number of public keys at each node is also 'x'.

To find the values of 'x' and 'y', an algorithm has been mentioned where the following steps are to be done.

Derivation of 'x' and 'y':

I=2

While(C (I, floor (I/2)) <no. of nodes)

I=I+1,

where 'x'=I and 'y'=floor (I/2)

Let number of nodes, n is 200 and initialize l=2. We can substitute values and calculate 'x' and 'y' as shown below. Consider n=200, I=2, taking the calculations, C(2,1)=2 < 200 so, I=3; C(3,1)=3 < 200 so, I=4; etc and finally we get 'x'=10 and 'y'= 4. Here, in a network with 200 nodes, each node needs to have 'x+y' keys, i.e., (10+4)=14 keys. In the traditional public key cryptography, a node in a network of size 200 should store (200+1)=201 keys (200 public keys and its own private key). This is how we achieve the memory efficiency which is one of the objective of the scheme and the main advantage of using this scheme.

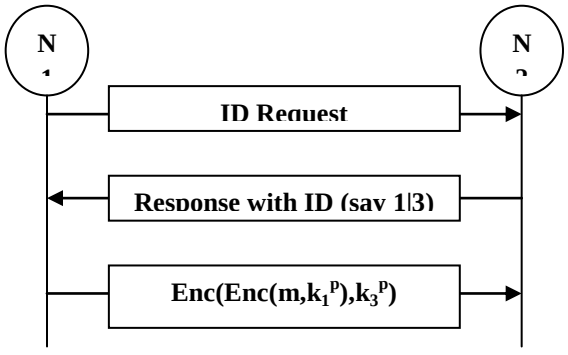

Fig.1 Secure communication using SMOCK

Table 1. An example of private key allocation

User	Private Keys at each User	Node ID
1	$K_1^R=\{k_1^r,k_2^r\}$	1|2
2	$K_2^R=\{k_1^r,k_3^r\}$	1|3
3	$K_3^R=\{k_1^r,k_4^r\}$	1|4
4	$K_4^R=\{k_1^r,k_5^r\}$	1|5
5	$K_5^R=\{k_2^r,k_3^r\}$	2|3
6	$K_6^R=\{k_2^r,k_4^r\}$	2|4
7	$K_7^R=\{k_2^r,k_5^r\}$	2|5
8	$K_8^R=\{k_3^r,k_4^r\}$	3|4
9	$K_9^R=\{k_3^r,k_5^r\}$	3|5
10	$K_{10}^R=\{k_4^r,k_5^r\}$	4|5

The key allocation is as follows: Each node will be given all the 'x' public keys and a combination of 'y' private keys so that none of the combinations are same and no key will get repeated in a combination. Consider an example of a small group with 10 users. We need 5 distinct public-private key pairs to build pair-wise secure communication channels among 10 users, which can be (K_1^r; K_1^p), (K_2^r; K_2^p), (K_3^r; K_3^p), (K_4^r; K_4^p), (K_5^r; K_5^p). Each user keeps 5 public keys and 2 private keys. The unique private key set allocation for each user for this scenario is shown in Table 1. The ID is used while communication as what shown in Fig.1 where secure communication is done between 2 nodes.

The main advantage of this scheme is that it reduces the number of cryptographic keys that are used in the network using the combinatorial design of key management scheme. The following graph in Fig.2 shows the actual difference in the number of keys stored in each node.

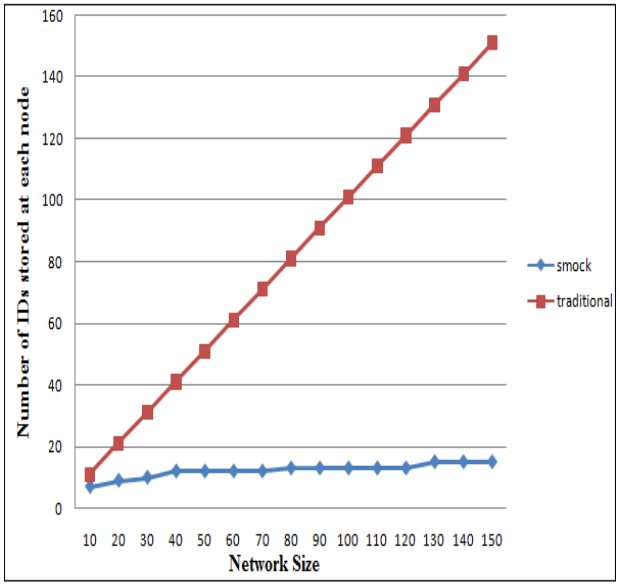

Fig.2 Comparison of Traditional and SMOCK schemes (With respect to Memory Efficiency)

The above graph shows the memory efficiency that can be achieved by SMOCK scheme and thus it can be used for networks with users having less memory and energy. With this we can say that SMOCK can be used effectively for Wireless Sensor Networks which have less amount of memory and that is being discussed in the following parts.

3. ADVERSARY MODEL/PROBLEM STATEMENT

When we use the SMOCK scheme of key management, what it does is just decreasing the number of nodes and providing integrity and confidentiality with the use of node ID. But when we use this scheme, the mentioned integrity and confidentiality will be achieved only after getting the correct node ID and start the secure communication. It is important to check whether the ID which is received is the actual node's ID or whether it is sent by an attacker or another node impersonating the actual node. For example, consider 'node 1' sends a request to 'node 2' which is been overheard by a nearby 'node 3'. If 'node 3' is the attacker who masquerades or impersonate as 'node 2' and sends his own information to 'node 1' which thinks that the data came from 'node 2' and continues the communication. 'Node 2' may send its information later but will be rejected as the reply has reached the destination already, but from a false user, 'node 3'. The normal security features will not be enough for these kinds of attacks. That is, there should be some other means to authenticate the nodes and this is why in this paper we are going to enhance the existing SMOCK scheme by adding an authentication scheme with it and hence providing more security.

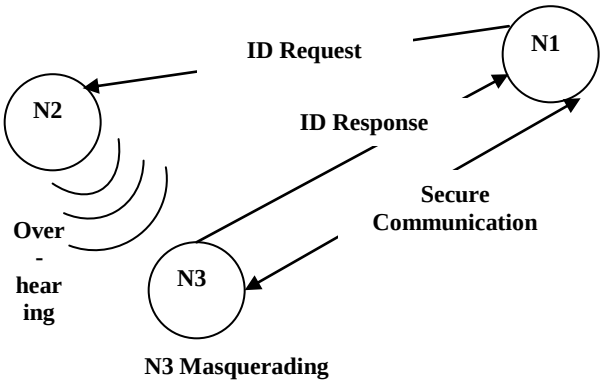

Fig.3 Attack Scenario

4. A-SMOCK: Authenticated SMOCK

A-SMOCK is the enhanced version of SMOCK where authentication is facilitated along with secure communication and memory efficiency. In the various authentication schemes considered in [9], the processing power needed and storage memory are a bit large. But, as authentication is very important and considering the above scenario, the problem with the existing scheme, Fig.3 draws us into two conditions where 1)The attacker is a dumb user and hence sender get responses from both the attacker and the valid user or 2)The attacker is an expert who jam either the request sent to or the response from the valid user and sometimes interrupt both. We consider both the cases and evolving some solutions for each ones.

4.1 Case I - Dumb Attacker

In the case of a dumb attacker, where attacker does not jam the request or response, the message requesting ID from 'node1' will reach 'node2' also which will be hence answered by the recipient. In this case, 'node1' will get two responses, one from the original recipient and another from the attacker. Normally what happens is the response which came first will be used and others will be rejected, which usually results in rejection of the reply from original user. To solve this problem, we can employ a time window where we check for duplicate replies for a request and if we receive any, the communication can be dropped or tried again. This will eliminate the attacks from dumb attackers and provide security without increasing the memory usage. One fault with the mentioned scheme is that we cannot identify which of the received replies is from the original user or attacker; instead it just finds the presence of the attacker and avoids it. Also, this can be used only if the attacker is dumb and in other cases, we need to use the following scheme.

4.2 Case II- Expert Attacker

When the attacker is an expert one, he jams the request and response from the sender and receiver and generates his own messages and transmits. In cases like this we need to ensure an efficient authentication scheme in the network. Implementation of a very basic authenticity can be done by verifying the node IDs of each node before initializing a communication. This can be done only if the nodes know the IDs of other nodes in the network with whom they have to communicate. This can be done by storing the IDs of all nodes in each node in the network, say node1 should have the IDs of node2, node3…node10 and same is with all the other nodes. If there are 'n' nodes in a network, each node will have a table with 'n-1' entries, the IDs of other nodes, in their memory. This scheme is referred as A-SMOCK with Store ID – ALL i.e., with storage of IDs of all nodes in network.

The whole function will be as follows: The centralized server generates and distributes keys to each node and a node ID will also be allocated. IDs of nodes can be preloaded by the server or can be sent by each node after the nodes are been deployed. The former will be the good method as the latter one needs more energy for sending IDs to each node and storing. And whenever a node has to communicate with another one it sends its own ID and asks for that node's ID. Each node can verify the other one's authenticity by checking whether the ID that is stored in its memory for the particular node is same as what it just received from the other one. If both the values matches, node can assure that it is communicating with the original node and can continue a secure communication.

If we use this A-SMOCK with Store ID – ALL in WSN, it will be practical only for small scale networks where the number of nodes is very less. As the network size increases, the number of entries that should be stored in each node will increase and the memory used will be high thus overrule the memory efficiency attained by SMOCK scheme. So, we should find another authentication scheme which uses less memory and that can be used for large networks.

5. A-SMOCK WITH STORE ID-N

As already discussed, the ID storing method has the disadvantage of using more memory and thus we proposes the enhanced version of ID storage which uses less memory space. The whole method is similar to the previous one and the only difference is in the number of entries stored in each node. A very less number of entries will be stored at each node by which authentication can be done efficiently. This scheme will be much more appropriate for WSN as the storage space is very less.

This can be done by this improved method, A-SMOCK with Store ID – N, considering the IDs of neighbors alone of each node instead of storing the IDs of all the nodes in the network. As soon as the nodes are being deployed, each node should send its own ID to its neighboring nodes which indeed will be stored by them in their memory. Thus all the nodes will gather their neighbors' ID and whenever a node needs to communicate with another it asks for the ID as mentioned in SMOCK and will verify whether the ID it got matches with what it has in its memory and if both are same it can confirm that the node at the other end is a valid one.

This is explained in Fig.4 where 'node1' and 'node2' communicates. Let us take an example network and analyze the efficiency of this scheme, with a network of 25 nodes which are placed as grid, which is been shown in Fig.5.

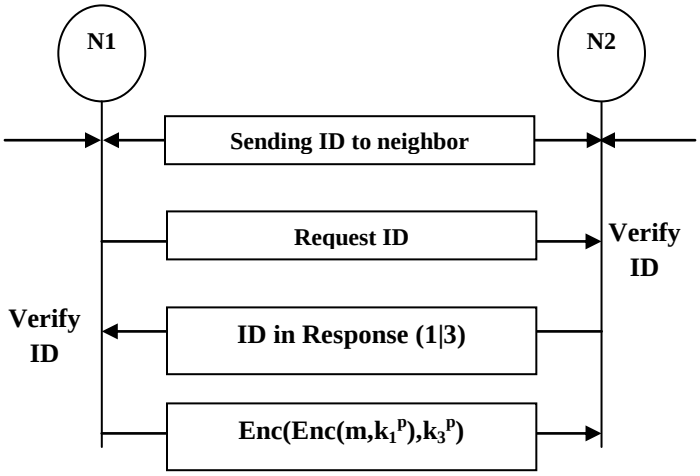

Fig.4 New 'Store ID-N' Model

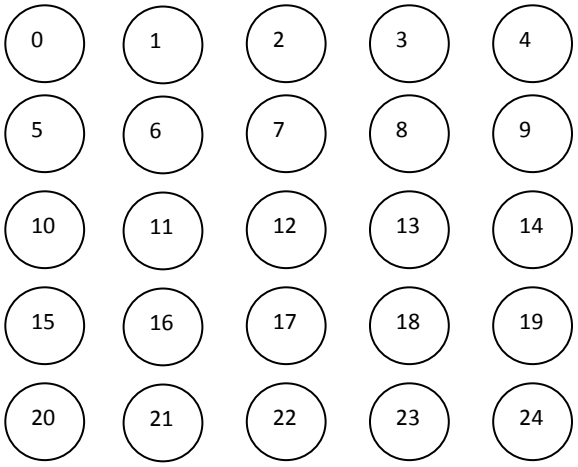

Fig.5 Nodes arranged in Grid

In a network with 25 nodes, the number of neighbors of each node will be the number of entries in them that means, the number of entries in every node will not be same. If arranged in grid format, 'node1' will have 3 neighbors, 'node2' will have 5 neighbors, 'node3' will have 5 and so on and this can be calculated for each network. Anyway, ultimate aim of reducing the number of entries and thus attaining memory efficiency with increased security ensured by adding authentication scheme is achieved in this scheme.

6. ANALYSIS

When comparing the 'Store ID-ALL' method with the 'Store ID-N' scheme, the latter one will be found utilizing less memory and energy which will be apt for WSN. An analysis of memory efficiency of both is illustrated in Fig.6. Here, nodes are assembled as a grid and each gets the IDs of its neighbors and stores the values in their memory. It is already mentioned that the number of entries will be less in the 'Store ID-N' scheme than the 'Store ID-ALL' scheme. In the latter case, if there are 'n' nodes in a network, each node should store 'n-1' entries, i.e., if there are 10 nodes, then each node should store 9 entries, if network size is 20, each node should store 19 entries in their memory. But in 'Store ID-N' scheme, the number of entries will be less, as we are storing only the neighbors' IDs. It is already explained that the number of entries will be different for each node. In the graph given, we have calculated the average of 'm' nodes using the following formula, and this will give an average value which has been plotted in the graph.

$$AverageNumberOfNodeIDs = \frac{\sum_{i=0}^{n} NumberOfNeighborsNode_i}{TotalNumberofNodes}$$

This graph shows exactly how much memory can be saved using the 'Store ID-N' scheme.

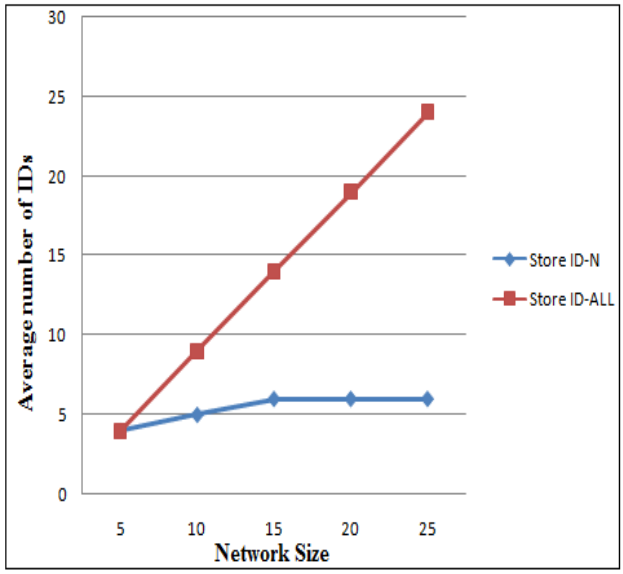

Fig. 6 Comparison of Store ID-ALL and Store ID-N schemes

7. CONCLUSION

In this paper the usability of SMOCK – Scalable Method of Cryptographic Key Management Scheme which is originally designed for Wireless Ad Hoc Networks, for solving the Key Management problem in Wireless Sensor Networks is examined and found to be a cost-effective protocol in terms of memory which is a scarce resource in WSN. But while trying to adopt the SMOCK for WSN, it brings out the new vulnerabilities in SMOCK like masquerading attacks because no explicit node authentication establishment phase is provided with SMOCK. To overcome this vulnerability three new approaches are proposed considering the different level of expertise of the

attackers. In further analysis of the three approaches, 'A-SMOCK with Store ID – N' approach performs well in terms of memory usage and provides better security. Thus we can infer that the 'A-SMOCK with Store ID – N' scheme is the better solution for Wireless Sensor Networks.

8. ACKNOWLEDGMENTS

Our thanks to the experts who have contributed towards the development of this paper.

9. REFERENCES

[1] Jennifer Yicka, Biswanath Mukherjeea, and Dipak Ghosa, "Wireless sensor network survey," Computer Networks: The International Journal of Computer and Telecommunications Networking, vol. 52 , no.12, pp. 2292-2330, August, 2008.

[2] Jaydip Sen, "A Survey on Wireless Sensor Network Security," International Journal of Communication Networks and Information Security, vol. 1, no. 2, pp. 55 - 78, August 2009.

[3] S. Kent and T. Polk, "Public-key infrastructure (x.509) (pkix) charter." [Online].Available: http://www.ietf.org/html.charters/pkixcharter.html.

[4] L. Zhou and Z. J. Haas, "Securing Ad Hoc Network", IEEE Network Magazine, vol. 13, no. 6, pp. 24-30, Nov. 1999.

[5] S. Capkun, L. Buttyan, and J. P. Hubaux, "Self organized public-key management for mobile ad hoc networks," IEEE Trans. Mobile Computing, vol. 2,no. 1,pp. 52-64, January - March 2003.

[6] Yang Xiao, Venkata K. Rayi, Bo Sun, Xiaojiang Du, Fei Hu, and Michael Galloway, "A survey of key management schemes in wireless sensor networks," Computer Communications In Special issue on security on wireless ad hoc and sensor networks, vol. 30, no. 11-12, pp. 2314-2341, September 2007.

[7] Wenbo He, Ying Huang, Ravishankar Sathyam, Klara Nahrstedt, and Whay C. Lee, "SMOCK: A Scalable method of cryptographic key management scheme for wireless ad hoc networks," IEEE Transactions on Information Forensics and Security, vol. 4, no. 1, pp. 140-150 , March 2009.

[8] S.A. Camtepe and B. Yener, "Combinatorial design of key distribution mechanisms for wireless sensor networks," In Pro 9th European Symposium on Research in Computer Security (ESORICS 04), pp. 346-358, 2004.

[9] Kai Zeng, Kannan Govindan, and Prasant Mohapatra, "Non-Cryptographic Authentication and Identification in Wireless Networks", in IEEE Wireless Communications, vol. 17, no. 5, pp. 56 - 62 , October 2010.