AN APPROACH FOR PROTECTING DATA INTEGRITY IN A SENSOR NETWORK

K. B. Neelima
Research Scholar,Bharath University
neelima.kanagala@gmail.com

ABSTRACT

Many applications that make use of sensor networks require secure communication. Because asymmetric-key solutions are difficult to implement in such a resource-constrained environment, symmetric-key methods coupled with a priori key distribution schemes have been proposed to achieve the goals of data secrecy and integrity. These approaches typically assume that all nodes are similar in terms of capabilities, and hence deploy the same number of keys in all sensors in a network to provide the aforementioned protections. In this paper, we demonstrate that a probabilistic unbalanced distribution of keys throughout the network that leverages the existence of a small percentage of more capable sensor nodes can not only provide an equal level of security but also reduce the consequences of node compromise. To fully characterize the effects of the unbalanced key management system, we develop, implement and measure the performance of a complimentary suite of key establishment protocols known as LIGER. Using their pre-deployed keys, nodes operating in isolation from external networks can securely and efficiently establish keys with each other. Should resources such as a backhaul link to a key distribution center (KDC) become available, networks implementing LIGER automatically incorporate and benefit from such facilities. Detailed experiments demonstrate that the unbalanced distribution in combination with the multi-modal LIGER suite offers a robust and practical solution to the security needs in sensor networks.

Key Terms: Heterogeneous sensor networks, probabilistic key management, probabilistic authentication, hybrid network security

INTRODUCTION

Sensors are inexpensive, low power devices, which have limited resources. They are small in size and have wireless communication capability within short distances. A wireless sensor network (WSN) is composed [2] of a large number of sensor nodes with limited power, computation, storage and communication capabilities. Environments, where sensor nodes are deployed, can be controlled and uncontrolled. If the environment is known and under control, deployment may be achieved manually to establish an infrastructure. If the environment is uncontrolled or the WSN is very large, deployment has to be performed by randomly scattering the sensor nodes to target area. It may be possible to provide denser sensor deployment at certain spots, but exact positions of the sensor nodes cannot be controlled. Thus, network topology cannot be known precisely prior to deployment.

Wireless nature of communication, resource limitation, lack of fixed infrastructure, unknown network topology prior to deployment, all these makes security [3] is a difficult issue in sensor network. Moreover, in some deployment scenarios sensor nodes need to operate under adversarial condition. Security solutions for such applications depend on existence of strong and efficient key distribution mechanisms. It is infeasible, or even impossible in uncontrolled environments, to visit large number of sensor nodes, and change their configuration. Moreover, use of a single shared key in whole WSN is not a good idea because an adversary can easily obtain the key. Thus, sensor nodes have to adapt their environments, and establish a secure network by using pre-distributed keys or keying materials, exchanging information with their immediate neighbors, or exchanging information with computationally robust nodes. Although there are ongoing works to customize public key cryptography [4] and elliptic key cryptography for low-power devices, such approaches are still considered as costly due to high processing requirements. Key distribution and management problem in WSN [5] is difficult one, and requires new approaches. One well-received solution that has been extended by several researchers is to distribute a certain number of randomly selected keys in each of the nodes throughout the network. Using this scheme, one can achieve a known probability of connectivity within a network. These previous efforts have assumed a deployment of homogeneous nodes, and have therefore suggested a balanced distribution of random keys to each of the nodes to achieve security.

Since a network consists of number of nodes having different capabilities, the concept of homogeneous system is not a practical one. Heterogeneity also provides new possibilities in terms of network connectivity. Here the concept of heterogeneity is used to provide more robust key management and establishment protocols for sensor networks. For that first we need a probabilistic unbalanced key management scheme. Unlike previous homogeneous key distribution schemes, the number keys stored by each node in the network is proportional to its inherent resources. Then derive probabilities for connectivity under a number of new key establishment trust models. Third, design and implement a suite of protocols call LIGER. Networks running the LIGER suite can not only establish keys between nodes in absence of a backhaul connection, but also take advantage of such resources when they become available.

UNBALANCED KEY DISTRIBUTION

Under the balanced approach [6], a large pool of P keys is generated, from which k are randomly selected, without replacement, for each sensor node. Two nodes may communicate to directly establish a session key if they have a key match. If nodes do not have a key match, they may still establish a session key through one or more intermediate nodes with which they each have a common key. In unbalanced approach, a key ring of size k keys are randomly selected from a key pool of size P and stored in each L1 node. Similarly a key ring of size m keys are selected and stored in each L2 node, where m >>k. The sizes of k and m can be varied according to the needs and constraints of the system designer. After key predeployment, nodes in the network discover the keys shared with their neighbors in the shared-key discovery phase.

The session-key establishment phase attempts to create a secure communication link for pairs of nodes within wireless transmission range that lack a shared key from the previous phase. If it is possible to communicate between a pair of nodes by using a multi-hop path through secure and trusted neighbors, then a key can be generated at the shared neighbor and distributed to the two endpoints. At the completion of this final phase, all nodes within transmission radius of each other should be able to communicate directly to whatever reliability the network designers have specified.

Chan [7] extended the basic scheme by requiring nodes to share at least q keys with each other. In so doing, it becomes more difficult for an attacker to compromise communications as q instead of one key must first be captured. While providing robustness to probabilistic keying, the q-composite scheme is not to consider for the purpose of node authentication. Specifically, if a node can prove its possession of multiple keys known to be assigned to it, its identity can be probabilistically authenticated. Under the balanced scheme, an increase in q leads to a significant increase in the number of keys stored by all L1 nodes. However, the unbalanced approach can reduce the burden of the q-Composite scheme on L1 nodes while retaining its security advantages.

KEY ESTABLISHMENT TRUST MODELS

There are mainly three trust models for key establishment: backhaul trust, peer-to-peer limited trust and peer-to-peer liberal trust. In backhaul trust model, the information collected by L1 nodes is to be backhauled to a resource outside of the local network. There is no need for an L1 to trust any node but its neighboring L2 node. If there is only one L2 node is present, then there must be a key match between the L1 and L2 node. If more than one L2 node is present in a neighborhood, two cases exist. In the first, all of the L2 nodes may act as gateways. In the second, only one L2 node will serve as a gateway to the backbone network. In this case, an L1 node may have a key match with the L2 gateway node, or because L2 nodes are trusted, may establish a session key through a path of one or more of the other L2 nodes.

In peer-to-peer limited trust, an L1 node desires to communicate with another L1 node in its neighborhood. To do this, it either requires a direct key match with the peer L1 node, or it must be able to establish a session key through a path of one or more L2 nodes. Because L2 nodes are assumed to be more secure due to the presence of key-protecting algorithms, all L2 nodes can be used.

In peer-to-peer with liberal trust to establish connectivity, L1 nodes may establish session keys using a path through a sequence of L1 nodes that provide mutual key matching. The above two models considered the use of up to $n-1$ hops for the establishment of a session-key. This has two drawbacks: First, the potential latency caused by allowing up to $n-1$ hops for initializing secure communication may be unacceptably high for the network to complete its mission, especially in cases of high mobility in which nodes may be required to establish new session keys often in the middle of active communication. Second, the introduction of multiple hops may increase the chance that compromised but undetected intermediaries are able to eavesdrop on the actual session-key establishment. While not in the direct path of future packet exchanges, compromised adjacent nodes that assisted with keying may still be able to overhear and decrypt the communications of uncompromised neighbors. Because each hop along an indirect keying path is able to decrypt the data from the previous hop, the probability that at least one compromised node is in a path increases as that path becomes longer.

PROTOCOL SPECIFICATION

The protocol for a network in an infrastructure-less environment will be referred to as LION. The scheme relying upon the presence of the KDC will be referred to as TIGER. LIGER [8] covers the integration of these two components. All nodes are loaded with a random set of keys drawn from a common pool before being deployed. The mapping of keys to nodes is stored in a KDC. If the network is operating in stand-alone mode, i.e., with no KDC, a protocol is designed to instantiate probabilistic keying. If the network has access to a KDC, the knowledge of the pre-deployed keys is used to perform probabilistic authentication with a high degree of confidence. The mode of operation may change between stand-alone and KDC-mode. In reality, large-scale sensor networks will have to optimally perform their missions in both modes. LIGER provides a robust method of key management for heterogeneous sensor networks. The combination enables different levels of probabilistic authentication without increasing memory requirements of the L1 sensor nodes.

Notations

A, B are principles (e.g. communicating L1 nodes)

ID_{A_0}, · · · , $ID_{A_{K-1}}$ are the sorted key identifiers corresponding to the keys held by node A.

K_A is a secret key known by node A.

$K_{A,B}$ is a session key shared between nodes A and B.

K_{A_AUTH} is an authenticator key for node A.

K_{A_i} is some key corresponding to an ID from within the range

L1 is a sensor node.

L2(GW) is the L2/Gateway node.

MAC(K_A,R|S) is a Message Authentication Code of the values R and S, using key KA

MAP_A is the bitmap corresponding to a sorted representation of ID_{A_0} , · · · , $ID_{A_{K-1}}$.

N is a nonce.

$\{S\}_K$ is a value S encrypted in key K.

LION: Standalone Key Management

After deployment under the LION protocol, an L1 node learns its neighbors through the exchange of Hello messages, and then attempts to establish keys with them. To accomplish this, the node broadcasts all of its key identifiers. Because the keys themselves are not transmitted and similar information could be gathered from a traffic analysis attack, this method does not compromise the integrity of the node itself. If a neighboring node overhears this transmission and determines that it shares one of the keys associated with the broadcast, it responds to the source with a challenge/ response.

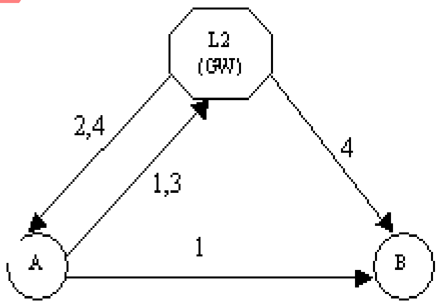

Fig: 2 LION key establishment

Messages 1 and 2 in Figure 4 show the message flow for the case in which a node, A, has a key match K_{A_i} , with a L2 node. The messages exchanged between the two exhibits the following format:

1. $A \rightarrow *: A, N, ID_{A_0}, \ldots \ldots, ID_{A_{K-1}}$
2. $L2 \rightarrow A : A, L_2, N, ID_{A_i}, \{ID_{A_i}, L2, N\}_{K_{A_i}}$

L1 nodes amass a list of neighbors with which they do and do not share keys. When the shared-key discovery phase ends, a node attempts to use the neighbors with which keys are already shared to assist it in establishing secure connections with all neighbors. The L2 node, having already established a link with the targeted L1, transmits a message to the requester and targeted node containing a session key encrypted in each of the keys shared with the L2 node. Each L1 node then receives the L2 broadcast, decrypts the session key and begins the secure transmission of data. The messages for the indirect phase are:

3. $A \rightarrow L2 : A, B, N$
4. $L2 \rightarrow: A, B, N, \{K_{A,B}, N\}_{K_{A,L2}}, \{K_{A,B}, N\}_{K_{B,L2}}$

If a node assists in establishing a session key during the indirect phase of the protocol, it deletes this key as soon as end-to-end communication is established. The two endpoints of communication also re-key immediately. In this way, if a node is compromised, it will not contain any valid session keys other than its own.

Probabilistic Authentication

One of the chief goals in the development of sensor network security is the minimization of memory overhead. Specifically, if the ability of an L1 node to perform its sensing task is limited by the memory footprint of a security solution, the security solution should be considered ineffective. Because one of the primary occupiers of memory in random pre-distribution schemes is the keys themselves, all efforts must be made to decrease this burden on the platform. Accordingly, LIGER only stores a single set of keys for use in both the LION and TIGER portions of operation. To provide robust operation in the face of disconnection with the KDC, these keys are pre-deployed as in the LION method. In order to prove its authenticity, a node instantiates a temporary authenticator key by which the system may perform probabilistic authentication. This key is created using a simple operation on a subset of the k pre-deployed keys in each L1 node. This scheme is also used for L1 nodes to loosely authenticate each other via an L2 node when a KDC is unavailable.

In TIGER, each L1 node uses q of its k pre-deployed keys to generate the authenticator key. The key itself is created by performing an XOR on the selected q keys. This operation is guaranteed to create an unguessable, pseudo-random value from the key space as demonstrated by Shannon. Because an attacker must know all of the key values associated with the creation of an authenticator key in order to derive it, this system is protected to a threshold of $q - 1$ for any given node. The hardness of breaking an authenticator key, for some $q < k$, is further enhanced as discussed in the protocol definition by allowing the subset of keys from which the authenticator key is derived to be changed as described below in the protocol specification. An analysis of the robustness of the authenticator key is given in Traynor [9]. Because the KDC knows all k keys pre-deployed in each sensor node, it can compute authenticator keys, and thus authenticate each L1 node in the network.

One drawback of the original q-composite method is that it requires an increase in the number of keys deployed in a L1 node to provide a reasonable probability of obtaining q key matches. When using a KDC in our system, no additional keys are required in the L1 nodes to maintain the likelihood of q key matches between a L1 nodes and a KDC because the KDC knows all of the keys deployed in the L1 nodes. In addition, because with unbalanced key distribution only a small number of keys are deployed in L1 nodes, the likelihood of one L1 node having q keys in common with a second L1 node and thus being able to impersonate it, is infinitesimally small.

TIGER: KDC-Based Key Management

In this before the system is initialized, each L1 node is bootstrapped with the same set of k keys as with LION. L2 nodes share a public/private key combination with the KDC. To perform authentication each node creates an authenticator key. This key is created using a simple operation on a subset of the k pre-deployed keys in each L1 node. This scheme is also used for L1 nodes to loosely authenticate each other via an L2 node when a KDC is unavailable.

L1 - L1 Authentication and Key Establishment

An L1 node A wishing to establish a secure and authenticated session key with a neighboring L1, a node advertising itself as B, begins the process by creating a token. The token itself is the MAC of a series of values included in the initial packet - the principles involved in the exchange, a nonce, and a sorted bitmap of the keys used to create the current authenticator key. Upon receiving the token, the node believed by A to be B makes a decision as to whether or not it desires an authenticated connection with the node it believes to be A. For example, if B has low battery power, is already congested with large amounts of data from other neighbors or has judged node A to be compromised, it may not wish to establish a key with A and thus drops the request.

If B decides to set up an authenticated relationship with A, it includes the token sent by A with its own token in a message to an L2 node. The L2 node then forwards the packet on to the KDC. Generating the appropriate authenticator keys for both A and B determine the validity of the two tokens according to the sorted bitmaps of the identifiers corresponding to keys used to make each token. If both tokens are deemed legitimate, the KDC responds with a message to the L2 containing a copy of a session key encrypted in a new, randomly chosen authenticator key for both A and B. The message from the KDC will also contain a bitmap corresponding to each of the authenticator keys used to sign the session keys. Nodes A and B then receive a transmission from the L2 node, generate the appropriate authenticator keys to unlock the session key and begin communication.

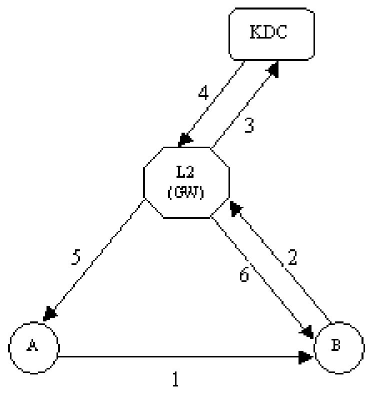

Fig: 3 TIGER flow for key establishment between L1 nodes

1. $A \rightarrow B : A, B, N, MAP_A, MAC(K_{A\_AUTH}, A|B|N|MAP_A)$

2. $B \rightarrow L2 : A, B, N, MAP_B,$
$$MAC(K_{B_{AUTH}}, A|B|N|MAP_B), MAP_A,$$
$$MAC(K_{A\_AUTH}, A|B|N|MAP_A)$$
3. $B \rightarrow L2 : A, B, N, MAP_B,$
$$MAC(K_{B_{AUTH}}, A|B|N|MAP_B), MAP_A,$$
$$MAC(K_{A\_AUTH}, A|B|N|MAP_A)$$
4. $L2 \rightarrow KDC : Forward\ Message\ 2\ to\ KDC$
5. $KDC \rightarrow L2 : A, B, N, MAP_{A'}, \{K_{A,B}, N\}_{K_{A'_{AUTH}}},$
$$MAP_{B'}, \{K_{A,B}, N\}_{K_{B'\_AUTH}}, MAC(K_{A,B}, A|B|N|K_{A,B})$$
6. $L2 \rightarrow A : A, B, N, MAP_{A'}, \{K_{A,B}, N\}_{K_{A'_{AUTH}}},$
$$MAC(K_{A,B}, A|B|N|K_{A,B})$$
7. $L2 \rightarrow B : A, B, N, MAP_{B'}, \{K_{A,B}, N\}_{K_{B'_{AUTH}}},$
$$MAC(K_{A,B}, A|B|N|K_{A,B})$$

L2 - L1 Authentication and Key Establishment

An L2 node wishing to view data collected by an L1 node must broadcast Hello messages in order to alert the L1 nodes of its presence. The L1 node A provides the L2 node with a token/message authentication code (MAC) created in the same way as described above; the L2 node forwards the contents of this message on to the KDC. If the KDC is able to verify the MAC, the L2 node will have verified that it is indeed in contact with node A. The KDC then returns a message to the L2 node containing a session key and a copy of the session key encrypted with the authenticator key of A. This information is included in a response to A, which also contains a MAC of the packet contents calculated with the KDC-generated session key. The last MAC can be confirmed as having been created by the L2 node after A has decrypted the session key. Because the L2 node cannot establish keys with the other remaining nodes in the network without their direct participation, least privilege is preserved

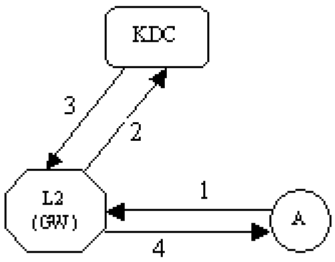

Fig: 4 TIGER flow for key establishment between L1 and L2 nodes

1. $A \rightarrow L2 : A, L2, N, MAP_A,$
$$MAC(K_{A\_AUTH}, L2|N|MAP_A)$$
2. $L2 \rightarrow KDC : Forward\ Message\ 1\ to\ KDC$

3. $KDC \rightarrow L2 : A, N, MAP_{A'},$

$$\left\{K_{A,L2}, N, \{K_{A,L2}, N\}_{(K_{A'\_AUTH})'}\right\}_{KDC,L2}$$

4. $L2 \rightarrow A : A, N, MAP_{A'}, \{K_{A,L2}, N\}_{K_{(A'_{AUTH})}},$

$$MAC\left(K_{A,L2}, A|N|MAP_{A'}, \{K_{A,L2}, N\}_{K_{(A'\_AUTH)}}\right)$$

LIGER

The combination of slightly modified versions of LION and TIGER schemes results in LIGER - a more robust method of key management for heterogeneous sensor networks. The combination enables different levels of probabilistic authentication without increasing memory requirements of the L1 sensor nodes. The advantage of LIGER is that it allows a sensor network to operate in a secure and efficient manner regardless of the available resources.

There are a number of tradeoffs experienced by a system operating in either mode. For example system likely to initialize using TIGER and transition into the LION protocol is a sensor network that is deployed in support of a planned operation. In this case, session keys [10] may be initialized in a controlled environment with access to a KDC. It is possible that access to the KDC will lost. In the ideal setting, a sensor network is allowed to initialize in the presence of KDC. Every node in the network is able to authenticate each of its neighbors to the full extent supported by this system. Because the KDC knows all of the keys stored in both the L1 and L2 nodes, it can send the key identifiers common to an L1/L2 pair to an L2 node in message 4 of the TIGER protocol flow. If the system later transitions in to the LION protocol, the stored, authenticated key identifiers now in the L2 node can be used for performing authentication of L1 nodes when refreshing expired keys or helping to establish an authenticated connection between two L1s without the presence of a KDC. The main limitation of this mode of operation is that in many cases the number of keys that L2 node knows will be small. If the L2 node is deployed with a much larger number of keys, the number of keys it has in common with L1 nodes may be much higher. The benefit of this approach would be that the authentication would be performed locally and the network delay incurred by accessing a KDC would be removed. A benefit of using the initial KDC connectivity to inform L2 nodes of the keys deployed in L1 nodes is that L1 nodes will no longer be required to broadcast their key IDs in stand-alone operation to establish session keys with L2 nodes. This reduces information leakage from the system.

A network operating in the military scenario suggested in the introduction would likely begin secure operation via the LION protocol. Because of the lack of friendly troops with connections to a backbone network and the need for a rapid deployment, initial access to a KDC may not be possible. The difficulty with the LION scheme, while providing security in the absence of a KDC, is that its ability to truly authenticate nodes is more limited. Some level of authentication is possible here that the initial broadcast of key identifiers is conducted during a network-bootstrapping phase wherein all nodes are free from compromise. During this bootstrapping phase, nodes can create a list of key identifiers present in each node. After that period, authentication of new connections can be achieved via the comparison of keys known to be shared with a neighbor versus those used to sign or encrypt a message.

It is always advantageous to initialize a network using TIGER. In this mode nodes may be authenticated with a confidence level, as an attacker was required to guess a 128-bit key. Also information may be securely distributed to L2 nodes so that some level of authentication may be performed if the KDC becomes unavailable. If the network must initialize using LION, some level of authentication may still be performed, but some information may be leaked to adversaries.

CONCLUSION

This paper leveraged the emerging trend of heterogeneity in sensor networks to provide new, more efficient mechanisms for secure communications. We began by introducing the probabilistic unbalanced key distribution. In this scheme, nodes with more intrinsic resources are responsible for a greater proportion of the communications and memory overhead associated with security. Second, we proposed a number of trust models for key establishment. The combination of these two techniques allows for the administrator of a system to implement policy at a much finer granularity and lower cost than in previous systems. Finally, we designed a multi-modal key establishment protocol. Whether in the presence or absence of a backhaul link, nodes running the LIGER suite are able to take advantage of all available security resources in potentially dynamic environments.

REFERENCES

1. Patrick Traynor, Raju Kumar, Heesook Choi, Guohong Cao, and Thomas La Porta, "Efficient Hybrid Security Mechanisms for Heterogeneous Sensor Networks" *IEEE transactions on Mobile Computing*, Vol.6, No.6, June 2007
2. I. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "A Survey on Sensor Networks," *IEEE Comm. Magazine*, Aug. 2002.
3. Michael Krishnan, "Intrusion Detection in Wireless Sensor Networks" *IEEE Conference*
4. William Stallings, "Cryptography and Network Security" 3rd edition, Pearson Education
5. R. Needham and M. Schroeder. Using encryption for authentication in large networks of computers. *Communications of the ACM*, 21:993–999, 1978.
6. L. Eschenauer and V. Gligor. A key management scheme for distributed sensor networks. *In Proceedings of ACM Conference on Computer and Communications Security (CCS)*, November 2002.
7. H. Chan, A. Perrig, and D. Song. Random key predistribution schemes for sensor networks. *In Proceedings of the IEEE Symposium on Security and Privacy (S&P)*, 2003.
8. P. Traynor, R. Kumar, H. B. Saad, G. Cao, and T. La Porta. LIGER: A Hybrid Key Management Scheme for Heterogeneous Sensor Networks. *In Proceedings of the ACM/USENIX Fourth International Conference on Mobile Systems Applications and Services (MobiSys)*, 2006.
9. P. Traynor, J. Shin, B. Madan, S. Phoha, and T. La Porta. Efficient Group Mobility for Heterogeneous Sensor Networks. *In Proceedings of the IEEE Vehicular Technology Conference (VTC)*, September 2006.

10. P. Traynor, H. Choi, G. Cao, S. Zhu, and T. La Porta. Establishing pair-wise keys in heterogeneous sensor networks. *In Proceedings of IEEE INFOCOM*, 2006.